FACULTAD DE ESTUDIOS ESTADÍSTICOS

MÁSTER EN MINERÍA DE DATOS E INTELIGENCIA DE NEGOCIOS

Curso 2019/2020

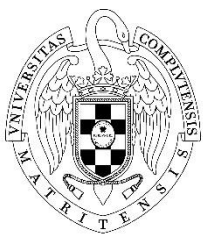
Trabajo de Fin de Máster

TITULO: Aplicación de técnicas de minería de datos en el mundo del tenis

***Alumno*: Alejandro Ruiz Hurtado**

***Tutora*: Aida Calviño Martínez**

Julio de 2020



UNIVERSIDAD COMPLUTENSE
MADRID

Resumen

El uso de los grandes datos y la estadística avanzada en el deporte es relativamente reciente. Si enfocamos la vista al mundo del tenis, se encuentran pocos precedentes. En este trabajo, se propone realizar un modelo de predictivo basado en algoritmos de Machine Learning para saber la duración de un partido de tenis antes de que se dispute, para facilitar la organización de los torneos y las parrillas televisivas. En segundo lugar, se propone realizar un modelo de clasificación que permita explicar qué influye en la victoria o la derrota del tenista español Rafael Nadal.

Palabras clave: Analítica del deporte, Tenis, Machine Learning, Predicción de la duración, Rafael Nadal

Abstract

The usage of big data and advanced statistics in Sports is relatively recent. If we look at the world of tennis, there are few precedents. In this paper, we build a predictive model based on Machine Learning algorithms for the duration of a tennis match before it is played, to facilitate the organizational support of tournaments and television schedules. Secondly, we develop a classification model to explain what influences the victory or defeat of the Spanish tennis player Rafael Nadal.

Keywords: Sport Analytics, Tennis, Machine Learning, Duration Prediction, Rafael Nadal

Índice de contenidos

1.Introducción

1.1 Definición del caso a investigar

La industria deportiva es una de las industrias más rentables, no solo a nivel nacional, sino en todo el mundo (El Español, 2018). Entre los deportes más populares nos encontramos con el tenis. Por poner un ejemplo, gracias a la Copa Davis celebrada el pasado noviembre se estimaron 40 millones de euros de impacto en la economía madrileña. Según un estudio de la Universidad Europea de Madrid (Universidad Europea, 2019), la Mutua Madrid Open celebrada en 2017 generó 107 millones de euros. Los dos torneos anteriores son solo casos concretos a nivel nacional, pero sirven de ejemplo para apreciar las enormes cantidades de dinero que mueven eventos de este tipo. Más allá del dinero generado, las competiciones generan también de manera indirecta un gran beneficio para el lugar donde se lleva a cabo: generación de puestos de trabajo, inversión en los transportes públicos y privados de la ciudad, restauración y estancia de los asistentes y participantes del torneo.

Tanto el éxito como el prestigio de un campeonato dependen en gran medida de su organización. Existen numerosos factores que se han de tener en cuenta y que diferencian un torneo de otro: tipo de pista, número de participantes, sets necesarios para llevarse la victoria, número de recintos en los que se pueden celebrar partidos simultáneos, etc. Se hace, por tanto, necesario estudiar cada una de las variables que influyen en el desarrollo del torneo de cara a la organización de los horarios de juego. Sin embargo, a diferencia de otras modalidades deportivas, los partidos de tenis no tienen una duración establecida y, por ende, la misma depende de muchos factores. Podemos encontrarnos tanto con partidos que duran apenas una hora como con partidos que han llegado a durar más de 8. El ejemplo más reciente lo podemos encontrar en el partido que protagonizaron en el torneo de Wimbledon en el estadounidense John Isner y el francés Nicolas Mahut, con una duración de 11 horas y 5 minutos. Esta duración fue tan extremadamente inusual que el partido tuvo que celebrarse en tres jornadas diferentes, entre el 22 y 24 de junio del año 2010. Esta impredecibilidad de los datos dificulta la organización de los diferentes partidos de un torneo, teniendo que dejar siempre suficiente margen para poder asegurar que no es necesario aplazar o modificar el resto de los partidos. Pero ¿y si fuese posible aproximarnos a un modelo que nos ayudase a predecir la duración de los partidos de tenis? Podría suponer un aumento de los beneficios relativos al ahorro temporal y, en consecuencia, económico resultante de una organización excelente. En una primera parte de este trabajo se tratará de dar una aproximación a esta cuestión.

No obstante, más allá de los beneficios obtenidos por parte de los organizadores, es importante valorar la importancia que puede suponer obtener unos resultados concluyentes en relación a las variables que influyen en la duración para los propios tenistas, de cara a la preparación física y mental de los mismos.

De igual manera, hay otros agentes que podrían estar interesados en conocer con mayor exactitud cuánto va a durar un partido. No son muchas las casas de apuestas que se arriesgan a introducir la modalidad de juego de la duración de un partido de tenis (Web Apuestas, s.f.). Las casas de apuestas que recogen este formato se centran en aspectos como el tipo de superficie (un partido en tierra podría

llegar a durar más por número de peloteos), números de sets que son necesarios para ganar un partido, o quiénes son los jugadores que disputan el juego. Las casas de apuestas encuentran un aliado común a la hora de intentar predecir la duración de un partido de tenis: las televisiones. No son pocas las cadenas que han tenido o tienen problemas en cuadrar su oferta televisiva con la retransmisión de partidos de índole tenística, al no saber con exactitud cuándo va a acabar (Puntodebreak, 2012). Este problema se acentúa cuando el partido en cuestión es una final y atrae a una audiencia mucho mayor.

El análisis de las estadísticas que se pueden obtener de un partido de tenis también puede dar pie a otros estudios que pueden ser muy interesantes si se individualizan por jugador. En este trabajo, también se hará un estudio de las estadísticas que debe tener en cuenta un jugador a la hora de preparar un partido utilizando el propio partido como base. Para el estudio, se ha personalizado en Rafael Nadal, pero se podría hacer con cualquier jugador del mundo con un historial de partidos jugados suficientemente extenso.

Para ambas líneas de investigación se usará un conjunto de datos extraído de la página web *www.tennisabstract.com*. Este sitio web ofrece estadísticas completas de todos los partidos de tenis celebrados en cada año. Para este estudio, se ha focalizado desde el año 2013 hasta 2018, para el conjunto de datos de predicción de la duración, y desde 2011 a 2018 para el conjunto de datos de Rafael Nadal. Estos datos contienen todos los partidos que se han jugado en ese período de tiempo.

1.2 Estado del arte

La analítica en el deporte tiene una historia relativamente joven. En la NBA (la principal competición de baloncesto del mundo) no hace más de 10 años que se empezó a usar analítica avanzada para mejorar su rendimiento en el terreno de juego. Tuvo que ser Daryl Money, General Manager de los Houston Rockets, uno de los primeros impulsores en incluir este tipo de estudios (Abbas, N. M. 2019). Hoy en día, todos los equipos de la liga estadounidense tienen departamentos especializados en estadística. Pero no solo entidades puramente deportivas se han interesado en el tema, sino que grandes empresas están empezando a realizar análisis deportivos. El ejemplo más reciente se puede encontrar en Telefónica que, a través de LUCA, su división especializada en Big Data, realiza analítica de deportes como ciclismo, baloncesto, running o e-sports (LUCA, 2019). También se han sumado universidades de todo el mundo a este campo, en los que a través de diferentes workshops y talleres se realizan publicaciones y conferencias referentes a la analítica en diversos deportes (KU Leuven, 2020).

Si nos remontamos algo atrás en la historia, es un hecho que uno de los objetivos principales ha sido elaborar modelos predictivos con el fin de obtener beneficios mediante el juego. La premisa de este tipo de investigaciones siempre ha sido clara: si se consiguen datos históricos de todos los participantes se puede intentar predecir los resultados a futuro. En esta línea de investigación, y relacionándolo específicamente con el tenis, han ido encaminados trabajos fin de máster como el elaborado en esta propia Facultad (Nieto Agudo. T, 2015) en el que se aplican técnicas de minería de datos a resultados de partidos de tenis en

conjunción con las cuotas reales en casas de apuestas. Otro estudio relevante en 2018 fue llevado a cabo por el *Tinbergen Institute* (Gorgi, P., Koopman, S. J., & Lit, R. 2018) en el que los autores encontraron resultados interesantes a la hora de predecir resultados, así como elaborando *clusters* de jugadores según tipos de superficie.

Centrándonos en los temas a tratar en este trabajo, se encuentran algunos antecedentes interesantes. Ya en 2017 se llegó a una primera aproximación a la duración de los partidos del torneo de Monte Carlo usando modelos de regresión en R y en Python. (Kovalchik, S., & Ingram, M. 2018). En este estudio, los autores se centran tanto en el tiempo de juego en sí como el que transcurre entre punto y punto y en los descansos. Anteriormente, en 2014, ya se analizaron los efectos que tiene jugar en pista dura y tierra batida en la salud de los jugadores, midiendo factores como la presión sanguínea y el ritmo cardíaco, siendo influyentes a la hora de medir la duración del partido (Martin, C., et al, 2011).

Otro estudio de la Universidad de Jaén (Luque, 2014) se centra en el tiempo total de un partido de cara a la recuperación física de los jugadores para asentar una base de ayuda a los entrenadores con el objetivo de preparar mejor los entrenos.

Por último, fijando la vista en el estudio individualizado por jugadores, la empresa SAP ha anunciado en octubre de 2019 una nueva actualización para su sección analítica de deporte, en la que, tras un acuerdo con la WTA (Women Tennis Association), la asociación de tenis femenino más importante a nivel mundial, los entrenadores podrán acceder a datos en tiempo real del desempeño de sus jugadoras en el partido (SAP, 2019).

2.Objetivos

Este trabajo tiene dos objetivos fundamentales. Por un lado, ser capaz de elaborar un modelo predictivo para la duración de partidos de tenis y, por otro lado, realizar un modelo explicativo del rendimiento de jugadores profesionales de tenis tomando como ejemplo al español Rafael Nadal.

Para conseguir estos objetivos fundamentales se deberán alcanzar una serie de objetivos específicos. Por un lado, para el modelo predictivo de la duración de partidos de tenis:

- Determinar qué variables afectan más a la duración de un partido profesional de tenis.

- Evaluar y verificar el poder predictivo que tienen las variables estudiadas anteriormente en la duración de los partidos, como el tipo de superficie, diferencia entre el ranking de los jugadores, tipo de torneo, etc.

- Realizar una comparación y análisis de los diferentes modelos obtenidos para evaluar su capacidad predictiva a través de sus diferentes métricas y estadísticos.

Por otro lado, para el modelo explicativo del rendimiento de Rafael Nadal:

- Observar qué variables afectan más a que el tenista español gane o pierda un partido.

- Clasificar las variables según su importancia para obtener los modelos explicativos.

- Evaluar los diferentes modelos obtenidos en base a su calidad y facilidad de interpretación.

Con este estudio se pretende acercar más al usuario y las organizaciones deportivas las diferentes técnicas para el estudio avanzado de las estadísticas en el mundo del tenis, a la par que servir de base para la creación de cualquier aplicación basada en datos para el análisis avanzado de este deporte.

3. Metodología empleada

El marco teórico sobre el que se va a realizar este trabajo se basará en la metodología SEMMA, la cual se divide en 5 fases que dan nombre a sus siglas (Calviño Martínez, A. 2019):

- Muestreo (Sample): Esta fase consiste en extraer una muestra lo suficientemente representativa de los datos.

- Explorar (Explore): Es esencial explorar los datos que hemos seleccionado para detectar posibles errores. En esta fase se hará además una visualización básica de las variables para poder realizar un análisis descriptivo de los datos.

- Modificar (Modify): En esta fase de modificación de los datos corregiremos los errores que se hayan detectado en la fase anterior. Además, se seleccionarán y transformarán las variables de cara a la modelización.

- Modelizar (Model): Se intentará encontrar modelos que predigan la variable objetivo.

- Evaluar (Assess): Una vez obtenidos todos los posibles modelos, se hará una evaluación para comprobar la calidad de los mismos y poder comparar entre las diferentes predicciones. En este sentido, cabe destacar que la evaluación se realiza sobre datos de prueba o test, es decir, datos que se separan del archivo original con el objetivo de evaluar el rendimiento de los modelos obtenidos, una vez se han ajustado los modelos con datos de entrenamiento o train.

Si bien en esta metodología pueden no intervenir todas las fases que la componen, en nuestro caso se hace necesario pasar por todas ellas para poder llegar a un buen resultado.

Para cumplir los objetivos anteriormente descritos en el apartado 2, se usarán distintas de técnicas de análisis. En concreto, se realiza una división de técnicas y medidas de evaluación que se explica en los subapartados *3.1* y *3.2* de este trabajo.

3.1. Metodología para el Objetivo I: Predicción de la duración de un partido de tenis

Para el primer objetivo de este trabajo, cuya variable objetivo es continua, se usarán las técnicas de k-NN, regresión lineal, redes neuronales, Random Forest, Gradient Boosting, XGBoost y ensamblado de modelos. Las medidas de evaluación que se usarán serán las siguientes:

- Error cuadrático medio o **ASE**. Viene dado por la siguiente fórmula:

$$ASE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

  donde y_i e $\hat{y}_i$ son los valores reales y predichos, de la variable objetivo respectivamente.
- Raíz del error cuadrático medio: simplemente es la raíz cuadrada del ASE. A esta medida se la conoce como **RASE o RMSE** (dependiendo del sofware utilizado) y se suele usar por su mayor facilidad de interpretación, dado que se encuentra expresado en las mismas unidades que la variable objetivo. Como estamos hablando en términos de error, interesa que esta medida sea lo más pequeña posible.
- Coeficiente de determinación (**R^2** a partir de ahora), que mide la capacidad de explicación de la varianza del modelo). La fórmula es la siguiente:

$$R^2 = 1 - \frac{SSE}{SST}$$

  Siendo SSE el ASE multiplicado por el número de observaciones "n" y SST la suma de cuadrados de los errores que se comenten cuando se usa la media como valor de predicción para todas las observaciones. Cuando el modelo es muy preciso, este coeficiente toma valores cercanos a 1.

3.2 Metodología para el Objetivo II. Explicación y clasificación para Rafael Nadal

Para el segundo objetivo de este trabajo (variable objetivo binaria) se usarán las técnicas de clasificación mediante árboles y regresión logística. Se opta por los árboles de clasificación y la regresión logística dado que el segundo objetivo es crear un modelo explicativo, y los algoritmos mencionados en el primer objetivo no permitan dicha explicación.

Como medidas de evaluación principales en los modelos construidos se usarán medidas como **la tasa de acierto** o la **tasa de clasificación errónea,** entre 0 y 1. Se calcula como el cociente entre la suma de verdaderos positivos y verdaderos negativos entre la suma total de observaciones clasificadas (en caso de tasa de acierto) o como el cociente entre la suma de falsos negativos y falsos positivos entre las observaciones clasificadas (en caso de tasa de fallos o de clasificación errónea). En caso de tasa de acierto, interesa que la medida esté lo más cercana posible a 1. Por el contrario, en caso la tasa de fallos, un buen modelo tendrá una tasa cercana a 0.

También se usará como estadístico de selección el área bajo la curva **ROC.** Esta área mide la sensibilidad obtenida con los modelos frente a 1-especificidad, es decir,

la tasa la de falsos negativos. En este sentido, un buen modelo tiene una ROC lo más cercana posible a 1, mientras que un mal modelo tiene una ROC cercana a o 0.5.

3.3 Remuestreo y optimización de modelos

Las técnicas de *remuestreo* son necesarias en cualquier modelo que se realice para que el azar no influya en los resultados obtenidos. En este sentido, se suele trabajar con dos técnicas, que se usarán por igual en este trabajo.

- **Training-Test repetido**: Consiste en dividir los datos en entrenamiento y en test. De este modo, se entrenan los modelos con la parte de *training* y se prueban con la de *test* con semillas aleatorias en repetidas ocasiones. Esta técnica es apropiada cuando la cantidad de muestras de la que se dispone es grande. En este trabajo, se asigna un 70% de los datos a *training* y un 30% a *test.*
- **Validación cruzada**: Esta técnica consiste en dividir el conjunto de datos en submuestras y construir el modelo con todas las observaciones menos las de una submuestra y evaluarlo a continuación con las observaciones de dicha submuestra excluida. Es preferible a training-test, especialmente cuando el conjunto de datos es relativamente pequeño, pero tarda más en ejecutarse. También se usará la **validación cruzada repetida**, que consiste en repetir este mismo proceso aleatoriamente las veces que se indique. Como regla general en este trabajo, se realizan 4 particiones en los datos, para ir probando con diferente número de repeticiones en caso de realizar validación cruzada repetida.

En varios de los modelos de este trabajo se usa la técnica de **Early Stopping**, que consiste en regularizar los datos de entrenamiento de forma que se evite el sobreajuste, es decir, que el modelo se ajuste demasiado a los datos de los que se dispone, de manera que no predice bien los datos de prueba. Esta técnica consiste en dividir los datos en conjuntos de entrenamiento y validación, y establecer un criterio de parada de la estimación cuando el error en los datos de validación empieza a aumentar.

Además, y por regla general, se usará una rejilla o *grid* en la que se fijarán varios valores de los parámetros de muchos de los modelos que se van a probar, con el objetivo de determinar el mejor modelo posible.

3.4 Algoritmos utilizados

3.4.1 k-NN

El modelo del vecino más próximo (más conocido por sus siglas en inglés, k-NN, de k nearest neighbours), es un modelo de predicción que se basa en la idea de que las observaciones con valores parecidos de las variables inputs, deberían tener valores similares también en la variable objetivo. En la Figura 1 se puede encontrar un ejemplo ilustrativo de clasificación con k-NN.

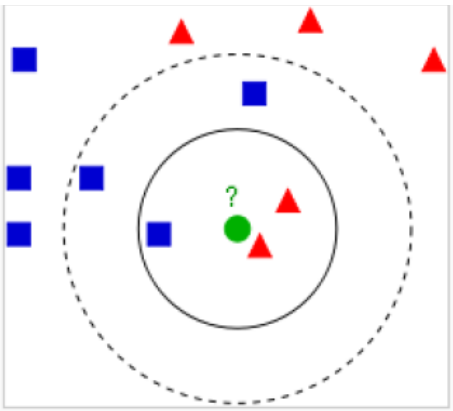


Figura 1. Ejemplo de clasificación con k-NN

De este modo, la predicción consiste en buscar las *k* observaciones más próximas a la observación que se quiere predecir, utilizando únicamente la información de las variables input, y se obtiene el valor de predicción de esta, a partir del valor de las variables objetivo en dichas observaciones. En nuestro caso, el valor de predicción que se obtiene viene dado por la media, ya que la variable objetivo que se va a tratar con esta técnica es continua.

El número óptimo de vecinos depende del conjunto de datos que se utilice en cada caso. Sin embargo, hay que tener en cuenta que conforme se vayan variando los valores de *k* (número de vecinos), también irán variando el sesgo y la varianza de los modelos. Lo habitual es probar con varios valores de *k* y seleccionar el que menor error produzca en los datos de prueba o test. La proximidad entre los *k* se mide a partir de la distancia entre las observaciones, siendo la distancia euclídea la que se utiliza habitualmente.

Para los modelos con k-NN, es necesario estandarizar las variables input de modo que todas tengan los mismos pesos a la hora de calcular la distancia. Si se cuentan con variables de clase o nominales, es necesario convertirlas a variable de intervalo, mediante la creación de variables *dummy.* Estas variables ficticias tomarán el valor 1 y si la observación correspondiente toma el valor de un determinado nivel de la propia variable y 0, en otro caso.

3.4.2 Regresión lineal

La regresión lineal es un modelo estadístico clásico que consiste en predecir una variable dependiente *y*, a partir de un conjunto de variables independientes x_i. La fórmula de la regresión lineal viene dada por:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \epsilon\,,$$

donde ϵ, es el residuo o el error que se comete en la estimación, y cada uno de los β son los valores de los parámetros que se obtienen en la predicción, y que miden la influencia que tienen las variables x_i sobre la variable dependiente y. Estos valores de β no se conocen a priori, por lo que será necesario estimarlos. Para conocer su valor, se buscará minimizar el error cometido por el modelo. Este error viene dado por la suma de cuadrados de los errores (SSE) explicada previamente.

Para los modelos de regresión lineal, es necesario que las variable cualitativas o nominales, entren en formato numérico. Para ello, se crean variables *dummy*, al igual que en otros métodos.

3.4.3 Regresión logística

La regresión logística es similar a la regresión lineal en cuanto al objetivo, que es el de predecir una variable dependiente *y* a partir de un conjunto de variables independientes x_i. Sin embargo, la regresión logística se usa cuando la variable objetivo que se quiere predecir no es continua, sino de clase. El modelo de regresión logística, para el caso de variable objetivo binaria, se representa de la siguiente manera:

$$p_1 = P(Y = 1|x_1, x_2, \ldots, x_m) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_m x_m)}}$$

Lo que implica que:

$$\log\left(\frac{p_1}{1 - p1}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_m x_m$$

El término de la izquierda recibe el nombre de *logit* y es el logaritmo del *odds ratio*. Este concepto viene a representar el cociente entre los *odds*, es decir, el cociente entre la probabilidad de que ocurra el suceso y la probabilidad de que no ocurra.

Para los métodos de regresión logística y lineal se utilizan métodos clásicos de selección de variables, a saber:

- **Selección hacia delante o Forward**: este método consiste en la introducción secuencial de variables en el modelo. La primera variable introducida es aquella que tiene más correlación con la variable objetivo. Las demás se introducen una a una hasta que no haya variables que mejoren el modelo. Una vez ha entrado una variable en el modelo, no puede volver a salir.
- **Selección hacia atrás o Backward**: este método consiste en, partiendo del modelo inicial que contiene todas las variables, ir eliminando una a una aquellas que menos significativas en el modelo hasta que queden dentro todas las que influyan en cierta manera.
- **Selección paso a paso o Stepwise**: este método es una combinación de los dos anteriores. De manera similar al Forward, comienza a introducir de manera paulatina las variables, pero en cada etapa se pueden eliminar las variables que han entrado el modelo. Esto tiene su explicación en que, al entrar una variable, puede hacer menos significativo el aporte de alguna otra.
- **Regresión LASSO:** Es un tipo de regresión que se puede ver como un método de selección de variables, ya que en cierto modo produce estimación de parámetros y selección de variables de forma simultánea. El método consiste en

limitar el número de parámetros del modelo de regresión clásico. Esto se debe a que, para ciertos valores de su parámetro de penalización, los coeficientes beta del modelo de regresión tienden a contraerse hacia 0 y algunos de ellos se anulan.

3.4.4 Redes neuronales

Las redes neuronales artificiales son algoritmos que imitan el funcionamiento del cerebro humano, donde cada neurona que procesa determinada información aprovecha las conexiones existentes con el resto de las neuronas para combinar información. Las redes neuronales están formadas por nodos interconectados, que forman diferentes capas de la red. Así, se tiene una capa input, que se puede conectar con una o más capas ocultas, que a su vez se conectan con la capa output (predicciones). En la Figura 2 se puede ver un ejemplo gráfico de red neuronal.

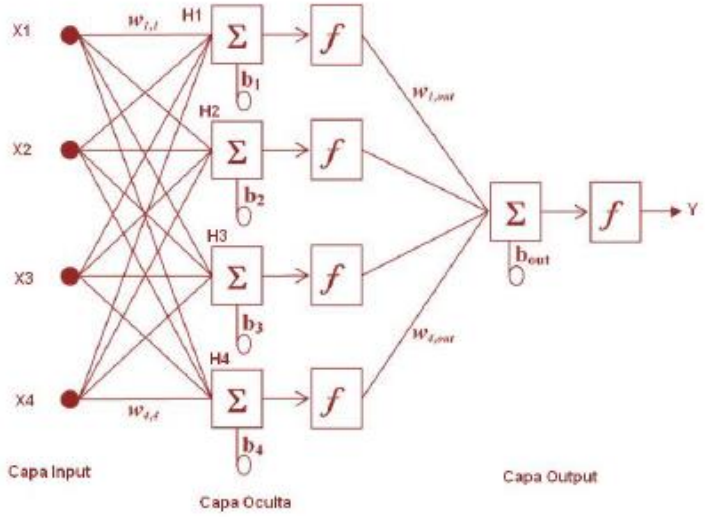


Figura 2. Ejemplo de red neuronal

Cada neurona es capaz de calcular una suma ponderada de sus entradas para luego aplicarle una función de activación, usada resolver problemas complejos derivados de la no linealidad entre las variables. Esta información será recogida por la siguiente neurona para repetir el mismo proceso. La conectividad entre neuronas la marcan los pesos, que hacen el papel de parámetros a estimar, de manera similar a como ocurría con la regresión lineal. Una de las grandes diferencias o ventajas de las redes con respecto a la regresión, es que el hecho de ponderar las sumas y multiplicarlas por una función de activación, hace que sean muy útiles para encontrar relaciones no lineales entre las variables. Los parámetros que se pueden configurar son los siguientes:

- *Size:* hace referencia al número de nodos, es decir, las unidades que se encuentran en la capa oculta.
- Función de activación: las más utilizadas son Tanh (tangente hiperbólica) o Softmax. En este trabajo, se utilizará por defecto Tanh y se indicará cuando se realice un cambio en la función de activación.
- Algoritmo de optimización: son algoritmos que se usan para mejorar la precisión de la red al estimar los parámetros, con el objetivo de minimizar la función de error. Los más comunes son *Levenberg-Marquardt* (LEVMAR), *Back Propagation* (BPROP) o *Quasi-Newton* (QUANEW)

- *Decay* de los pesos de la red o *learning rate*: es un parámetro que ajusta el algoritmo de optimización controlando la velocidad a la que se mueve dicho algoritmo hasta llegar al mínimo de la función de pérdida de error.

3.4.5 Árboles de decisión

Los árboles de decisión son una técnica muy útil para explicar o jerarquizar variables de una manera muy sencilla y sin asunciones teóricas previas sobre los datos. A partir de un nodo original o raíz que contiene todo el conjunto de datos, los árboles dividen la muestra en subconjuntos o nodos hijos, buscando una segmentación con la mayor homogeneidad posible.

Una vez se ha obtenido esta segmentación, se les asigna un valor de predicción a aquellos valores que no tienen sucesores (que reciben el nombre de hojas), de forma que todas las observaciones pertenecientes a dichas hojas serán predichas o clasificadas en torno a ese valor. En los árboles, se podrán ajustar parámetros como el número de ramas (se dejará en 2 por defecto dada la preferencia por árboles binarios), el número de observaciones por hoja (principal parámetro que mide la compleidad), la profundidad máxima o podar el árbol (eliminar hojas del modelo original para evitar el sobreajuste). También se podrán definir los criterios objetivo para la construcción del árbol. Para el caso de variables objetivo de clase, estos criterios son los siguientes:

- *Test de la χ^2(ProbChisq):* Se contrasta la hipótesis nula de que la variable "escogida" no influye en la variable objetivo a través de su p-valor. La idea es buscar variables que influyan de manera significativa en la variable objetivo.
- *Índice de Gini:* Este índice permite observar cómo de homogéneas son las observaciones de un nodo u hoja con respecto a una variable categórica. Un nodo "puro" tendrá un valor de 0 y uno con la misma proporción de eventos y no eventos tendrá un valor de 0,5. La fórmula de este índice cuando la variable objetivo es binaria se representa de la siguiente manera:

$$IG(nodo\ j) = 1 - \left(\frac{n_{ev}^j}{n_j}\right)^2 - \left(\frac{n_{no}^j}{n_j}\right)^2$$

- *Entropía:* Al igual que el índice de Gini, permite medir la homogeneidad de un nodo, si bien esto se mide mediante la función de entropía, a saber:

$$E(nodo\ j) = -p_{ev}^j \log_2(p_{ev}^j) - p_{no}^j \log_2(p_{no}^j), \text{ donde } p_{ev}^j = \frac{n_{ev}^j}{n_j} \text{ y } p_{no}^j = \frac{n_{no}^j}{n_j}$$

  Un nodo "puro" tendrá un valor de entropía igual a 0, mientras que un nodo con igual proporción de eventos y no eventos, tendrá un valor igual a 1.

En este trabajo, los árboles de decisión propiamente dichos cobran importancia en el segundo objetivo de este trabajo, que es la realización de un modelo explicativo. Para ello, se utilizarán árboles de clasificación al tener una variable objetivo binaria. Sin embargo, también son la base sobre la que se construirán técnicas de predicción avanzadas como *Bagging, Random Forest, Gradient Boosting y XGBoost*, que son también aplicables a variables objetivo continua (aunque en este caso los criterios objetivo serán otros) y, por tanto, al primer de los objetivos de este trabajo. Estos algoritmos tienen en común con los árboles los parámetros que ya se han comentado,

como el número mínimo de observaciones por hoja, número de ramas o profundidad máxima. Estos algoritmos solucionan las posibles desventajas que pueden tener los árboles combinando el resultado de muchos de ellos (añadiendo iteraciones) y añadiendo parámetros de regularización, que serán explicados en cada caso.

3.4.6 Random Forest

El algoritmo de *Random Forest* combina árboles, de modo que los valores de cada árbol dependen de un vector aleatorio que está muestreado de forma independiente y de forma que siga la misma distribución para todos los árboles que se construyen. En otras palabras, este algoritmo consiste en incorporar dos fuentes de variabilidad, que son el remuestreo de las observaciones y la aleatoriedad de las variables, usadas para segmentar los nodos de los árboles. El algoritmo de Random Forest consiste en (Portela, 2020):

1. Repetir m veces las siguientes acciones:
    - Seleccionar N observaciones con reemplazamiento de los datos originales.
    - Aplicar un árbol de la siguiente manera: En cada nodo, seleccionar *p* variables de las *k* originales y de las *p* elegidas, escoger la mejor variable para la partición del nodo.
    - Obtener predicciones para todas las observaciones originales N.
2. Promediar las *m* predicciones obtenidas en el apartado 1.

Este algoritmo tiene por ventajas el de encontrar relaciones no lineales y predecir mejor cuando existen muchas variables categóricas o interacciones ocultas, además de evitar el problema de variables predictores muy dominantes. En otras palabras, al añadir aleatoriedad en las variables que se utilizan, se obtienen árboles no tan parecidos unos de otros, por lo que se consigue reducir la varianza del modelo final.

Los parámetros que se ajustan en los modelos de Random Forest son:

- *mtry*: número de variables que se sortean en cada división del árbol.
- ntrees: hace referencia a las iteraciones del modelo (número de árboles que se van a promediar).
- node size: se refiere al número de observaciones que habrá como mínimo en una rama-nodo, es decir, al tamaño mínimo de la hoja.

3.4.7 Gradient Boosting

Este algoritmo es uno de los más potentes en términos predictivos, y uno de los más usados en la actualidad. Este modelo usa como base el algoritmo de *Random Forest*, con la diferencia de que las variables no se sortean por defecto, y a su vez añade nuevos parámetros Consiste en actualizar los pesos de las observaciones en función del descenso del error que se comete durante la predicción. Los parámetros del modelo son:

- *n.minobsinnode*: representa el tamaño mínimo de nodos finales, es decir, el mínimo de observaciones que debe tener cada hoja. Este es el principal parámetro que define la complejidad del árbol.
- n.trees: es el número de iteraciones o de árboles del modelo.
- *bagfraction*: este parámetro define el porcentaje de observaciones que se sortearán antes de cada árbol. Como regla general, se deja en 100%, que es su valor con su defecto, para luego valorar si es conveniente modificarlo para evitar un posible sobreajuste o ver su efecto en los resultados finales.
- *Shrinkage*: es el parámetro de regularización, es decir, la constante que se encarga de medir la velocidad del ajuste con respecto a la función de pérdida de error. Cuanto más alto sea, la velocidad de convergencia será mayor, pero el ajuste será menos fino. Al contrario, si se fija en un valor muy bajo, necesitará más iteraciones para converger, aunque se recomienda esta opción de forma general.

3.4.8 XGBoost

Este algoritmo es una modificación del *Gradient Boosting*, bastante usado en concursos de predicción o clasificación. La principal corrección que introduce este algoritmo frente al anterior es la inclusión de factores de regularización para reducir el sobreajuste. Esta regularización consiste en controlar los valores de los parámetros, de forma que no obtengan valores excesivamente grandes o pequeños. De este modo, la función que se crea para predecir estará normalizada.

Los parámetros para controlar en R son:

- *eta:* hace referencia al *shrinkage*, es decir, la velocidad de ajuste.
- *gamma:* es el parámetro de regularización, que penaliza las predicciones complejas que pudiera tener una hoja de un árbol.
- *min_child_weight:* número mínimo de observaciones que tendrán los nodos finales.
- *max_depth:* Profundidad máxima del árbol, es decir, la distancia entre el nodo raíz y la hoja más lejana.
- *subsample:* número de observaciones que se utilizan para construir cada árbol.

3.4.9 Ensamblado

El ensamblado de modelos consiste en construir predicciones derivadas de la combinación de los resultados de otros modelos. Esto consiste en recoger las predicciones obtenidas por los diferentes algoritmos y combinarlas, para después promediar sus resultados. Esta técnica intenta mejorar la predicción de los modelos partiendo de la base de que un modelo puede corregir a los demás, consiguiendo por regla general modelos con menor varianza. Como parte negativa, estos modelos son más complejos y difíciles de interpretar que los diferentes modelos por separado.

4.Fuente de datos

Los datos sobre los que se va a trabajar han sido extraídos de la página web *www.tennisabstract.com*. En esta web, dentro del *"Match Charting Project''*, se pueden encontrar datos en bruto tanto de partidos de tenis individuales como de rankings en una amplia selección de años. En este caso, los dos conjuntos de datos, tanto el de la duración de los partidos como el de Rafael Nadal, están fabricados a partir del mismo, con un trabajo previo antes de entrar en la fase de depuración que se comentará en cada caso.

Para conseguir los objetivos fundamentales que se han mencionado en el apartado 2, se deberán manipular los conjuntos de datos que se van a trabajar. Por un lado, dado que la fuente de datos ofrece la información año por año, se realizará la tarea de juntar los años de estudio en un mismo fichero. Acto seguido, se hará una separación de los datos. Por un lado, para el objetivo de predecir la duración de los partidos, nos quedaremos con un fichero que contenga los datos de todos los partidos disputados obviando las estadísticas del propio rendimiento de los jugadores en los partidos, para poder así predecir la duración de los mismos con antelación. En el otro fichero de datos se extraerán los datos de los partidos en los que ha participado Rafael Nadal. Una vez se tengan los dos ficheros separados se podrá aplicar la metodología para la correspondiente realización del proyecto.

5. Objetivo I: Predicción de la duración de un partido de tenis

Para este conjunto de datos se han extraído los datos correspondientes a todos los partidos de tenis de categoría masculina celebrados entre 2013 y 2018.

5.1 Trabajo previo

En este conjunto de datos se han eliminado en Excel las variables referentes al desempeño de los jugadores durante el transcurso de los partidos, ya que son datos que no podremos conocer antes de su transcurso. Esto incluye eliminar variables como el resultado del partido y todas las estadísticas individuales recogidas en el mismo. Además, se han añadido columnas que son resultado de transformaciones de las ya existentes en el conjunto de datos original. El motivo de estas transformaciones es que la información estaba recogida según el vencedor o el perdedor del partido. Como el resultado del partido no estará disponible a la hora de obtener nuestra predicción, se crean nuevas variables a través de combinaciones, máximos y mínimos en diferentes variables. Estas variables están marcadas con asterisco en la Tabla 1.

5.2. Descripción de las variables

Para el conjunto de datos de predicción, se usarán las variables de la Tabla 1.

Tabla 1. Descripción de las variables para el objetivo I.

Variable	Descripción
tourney_name	Nombre del torneo.
surface	Tipo de superficie (Pista, tierra, hierba, alfombra o desconocida).
tourney_level	Categoría del torneo. (A=Abierto, M = Masters, G=Grand Slams, F=Finales (ATP, Olímpicas o NextGen), D=Copa Davis).
month	Mes en el que se juega el torneo. Extraída de la variable fecha del dataset original.
best_of	Hace referencia al número de sets que es necesario conseguir para ganar el partido (5 o 3).
round	Ronda del torneo (octavos, cuartos, semifinales…).
minutes	Duración en minutos del partido.
min_age *	Edad del jugador más joven del partido.
max_age *	Edad del jugador más mayor del partido.
players_hand *	Combinación de las manos de los jugadores (RR=ambos diestros, LL= ambos zurdos, RL = uno diestro y otro zurdo, Desconocido= todo lo que no sea lo anterior).

min_rank *	Puesto en el ranking del jugador mejor clasificado en el ranking ATP antes el partido.
max_rank *	Puesto en el ranking del jugador peor clasificado en el ranking ATP antes el partido.
dif_rank *	Diferencia absoluta ente el ranking ATP de los jugadores. (max_rank-min_rank).
min_rank_points*	Puntuación en el ranking ATP del jugador con menos puntos antes del comienzo del partido.
max_rank_points*	Puntuación en el ranking ATP del jugador con más puntos antes del comienzo del partido.
dif_rank_points *	Diferencia absoluta entre los puntos en el ranking ATP de los jugadores. (max_rank_points – min_rank_points).

5.3 Exploración de datos en SAS Enterprise Miner

Después de la primera depuración en Excel, se importa el archivo a SAS Enterprise Miner (SAS EM) con un total de 15.121 observaciones y 16 variables. Se hace necesario hacer una depuración más profunda que nos lleve a gestionar incoherencias en los datos, así como datos atípicos o faltantes.

Se realiza un primer análisis estadístico de las variables de clase y las de intervalo que se tienen en el conjunto, con el objetivo de detectar posibles errores para su posterior corrección (Figura 3).

Variable	Rol	Media	Desviación estándar	No ausente	Ausente	Mínimo	Mediana	Máximo	Asimetría	Curtosis
dif_rank	INPUT	81.14959	147.4792	15121	0	0	40	2125	5.661936	44.46147
dif_rank_points	INPUT	1395.237	2132.611	15121	0	0	610	16641	3.12485	11.67246
match_num	INPUT	147.577	123.4127	15121	0	1	127	701	0.106321	-1.6364
max_age	INPUT	30.08194	3.405328	15121	0	0	30	46	-0.22313	0.383176
max_rank	INPUT	125.1453	175.8494	15121	0	0	81	2159	5.152913	35.34108
max_rank_points	INPUT	2195.096	2453.258	15121	0	0	1295	16950	2.805884	9.13397
min_age	INPUT	25.39865	3.508992	15121	0	0	26	37	-0.04927	-0.29091
min_rank	INPUT	43.9957	69.1135	15121	0	0	30	1411	9.266634	131.8655
min_rank_points	INPUT	799.8594	754.2934	15121	0	0	659	11850	4.637441	36.42688
minutes	TARGET	107.4313	42.44104	15121	0	1	100	1146	2.51109	37.08271

Figura 3. Resumen estadístico de las variables de intervalo

Como se puede observar en la Figura 3, no hay ningún valor ausente y no parece haber ninguna anomalía en las variables. Sin embargo, hay un apunte interesante que es digno de mención con respecto a la variable objetivo. Podemos comprobar que la duración mínima de los partidos es de 1 minuto. Tras hacer varias comprobaciones en Excel sobre el conjunto de datos inicial, se puede llegar a la conclusión de que la mayoría de los partidos de tenis necesitan al menos media hora de duración para llegar a un resultado coherente. De lo contrario, la victoria o la derrota se debe a retiradas antes de tiempo. Sin embargo, el simple hecho de jugar un partido en el torneo (ya sea lesionado o sin estar en plenas facultades para competir), reporta beneficios económicos para los jugadores en términos de premio de torneo, por lo que algunos partidos tienen la mínima duración para que los jugadores puedan recibir dichos premios. Como no se puede cuantificar ni juzgar el verdadero estado de salud de un jugador antes de empezar un partido con los datos disponibles, ni mucho menos los propios intereses económicos, se deciden dejar

todos los datos que figuran en el conjunto, para tener así en cuenta la variabilidad real de la variable objetivo.

En cuanto a las variables de clase (7 en total), se han detectado algunos errores. En la Tabla 1 del Anexo A.1 se muestran los niveles de todas las variables de clase, previo a las modificaciones que se explican a continuación.

Hay 12 niveles en la variable *players_hand* cuando se han definido 4, como se observa en la Tabla 1. Esto se debe a las distintas combinaciones de la concatenación y que habrá observaciones con una U de "Unknown". Este problema se tratará unificando todo lo que no sea RR, RL o LL en Desconocido, que llamaremos U.

En superficies, vemos que hay representadas *Hard*, *Clay*, *Grass*, *Carpet* y *None*. Como de alfombra no hay apenas representación y se asemeja en gran medida a pista dura, se agrupará en *Hard* (pista dura). Lo mismo se hace con *None*, que quiere decir que no se tiene información respecto al tipo de pista. Como es solo un 0,5% de los datos, se les asigna a la pista dura, que es la más común. Nos quedarán por lo tanto 3 niveles: Pista dura, tierra y hierba.

Con respecto a los meses, se ve que tanto noviembre y diciembre tienen muy poca representación. Es cierto que en la recta final del año se juegan muy pocos partidos debido a las vacaciones y fin de la gira ATP (vuelve a comenzar en enero con el Open de Australia). Con el objetivo de no perder información que puede ser importante en variables de clase que tienen muchos niveles, como puede ser el caso de *tourney_name*, se puede configurar SAS EM de modo que automáticamente agrupe estos niveles en categorías nuevas que tendrán características similares. A esta técnica se le llama "seleccionar variables de grupo" y se hará con las variables categóricas, inclusive las que se han corregido manualmente con el objetivo de crear nuevas variables con un número inferior de categorías y no perder información predictiva de variables que pueden ser importantes. En la tabla 2 del anexo se puede ver el resultado final del output del nodo.

En la Figura 4 se puede ver el resumen del número de niveles de las variables de clase después de aplicar las distintas correcciones llevadas a cabo en SAS EM. Se puede observar también que el sistema únicamente agrupa la variable *toruney_name* (el prefijo G_ hace referencia a la agrupación anterior), mientras que el resto se quedan como están.

Variable	Etiqueta	Tipo	Número de niveles	Ausente
G_tourney_name	Grouped Levels for tourney_name	N	5	0
REP_players_hand	Replacement: players_hand	C	4	0
REP_surface	Replacement: surface	C	3	0
best_of	best_of	N	2	0
month	month	C	12	0
round	round	C	9	0
tourney level	tourney level	C	5	0

Figura 4. Niveles resumidos para las variables de clase del conjunto de datos

Con respecto al tratamiento de los datos atípicos y los *missings*, se hace un estudio de posibles datos atípicos, que son datos a priori numéricamente distantes del resto de datos. Estos datos pueden tener gran influencia si los modelos no son robustos. Cabe destacar que la base de datos que se ha construido es muy completa, con la mayoría de los campos rellenos y correspondientes a estadísticas "normales" de partidos de tenis. Por ello, el estudio de los datos atípicos no da lugar a demasiados casos en los que pueda ser necesario su tratamiento. Aun así, en estos casos se decide dejar estos datos como faltantes. Con respecto a los datos faltantes, casi la totalidad de estos se generan tras el tratamiento de los atípicos, por lo que se imputan teniendo en cuenta la distribución de la variable.

5.4 Importancia de las variables

Después de haber tratado correctamente los datos atípicos y los correspondientes datos faltantes, se puede seguir explorando el conjunto de datos para ver qué variables afectan más a la duración de los partidos.

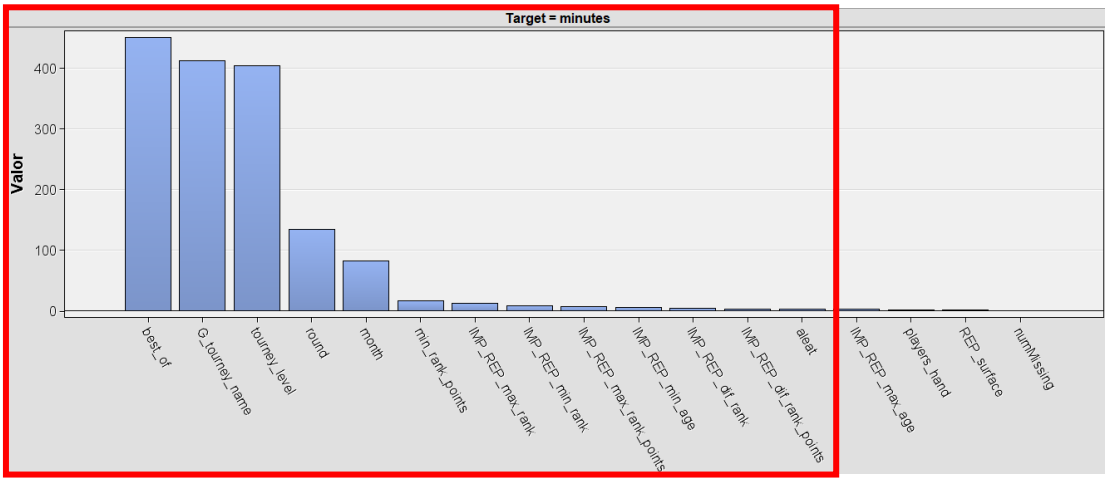


Figura 5. Importancia de las variables

En la Figura 5 se encuentra representado el valor[1] que tiene cada una de las variables input en términos de relación con la variable objetivo. En el gráfico, se puede observar cómo más allá de *best_of*, el agrupamiento que se ha hecho de *tourney_name* y el nivel del torneo que se expresa en *tourney_level*, no parece haber variables que influyan de sobremanera en la variable objetivo. La variable de intervalo que parece tener más importancia es *min_rank_points*. La variable *aleat* representa una variable aleatoria que se ha creado en la depuración para ayudar en la selección de variables. Así mismo, *numMissing* mide el número de missings que hay en cada observación. Es curioso observar cómo, a pesar de la creencia al comienzo del trabajo de que la superficie (*surface*) podría influir en la duración de los partidos de tenis, tiene ninguna importancia a la hora de determinar la misma (su poder predictivo es inferior al de la variable aleatoria).

[1] El valor es una medida calculada por SAS Miner que sirve para evaluar la relación de las variables inputs con la variable objetivo, tomando valores altos (no acotados) cuando la relación es fuerte y 0 en ausencia de la misma.

5.5 Selección de variables

La selección de variables juega un papel determinante en la modelización de cualquier conjunto de datos. Para ello, hay métodos como los de regresión o los basados en árboles que ya hacen su propia selección de variables, pero, para evitar que la predicción en el resto de las técnicas pueda verse perjudicada, se realizarán varios tipos de selección de variables.

Se llevará a cabo una primera selección de variables según la importancia de las variables. En la Figura 5 se ha visto la importancia de las variables input con la variable objetivo. El haber creado una variable aleatoria en la fase de depuración nos permite poder descartar todas aquellas que estén por debajo de la misma, es decir, que tengan menor poder predictivo. Así, quedarán fuera de la fase de modelización las variables *max_age, players_hand, surface y numMissing*. Una vez hecho esto, se eligen varias combinaciones de variables que serán introducidos a los modelos de predicción, en concreto:

1- Variables sin más procesamiento que lo acometido en los pasos anteriores.
2- Transformación de variables a través del método de correlación máxima.
3- Clusterización de variables de intervalo, es decir, se forman grupos de variables correlacionadas entre sí y se selecciona sólo una de cada grupo. De esta forma, se suprime información redundante.
4- Clusterización de variables de intervalo tras aplicar transformación de variables mediante el método de correlación máxima.

5.6 Modelización

Una vez se tienen las variables seleccionadas y transformadas para su mejor utilización en los modelos, se procede con las distintas técnicas de predicción explicadas anteriormente.

5.6.1 K-NN

Para el modelo con k-NN se han comparado las 4 combinaciones de variables anteriores con 5, 15, 25 y 35 vecinos cada uno (un total de 16 modelos), sobre un Training-Test con 10 repeticiones. Como ya se ha comentado previamente, para los modelos construidos bajo esta técnica, es necesario estandarizar las variables de intervalo y "dummificar" las de clase.

En la Figura 6 se puede observar que los modelos que menor error y variabilidad ofrecen son los 4 primeros, que coinciden con el primer camino que se ha mencionado. Este camino consiste en no aplicar ninguna técnica de clustering o transformación de variables. Aunque hay algunas iteraciones atípicas, de nada sirve tener otros modelos que no las tienen si tienen más variabilidad en el RASE.

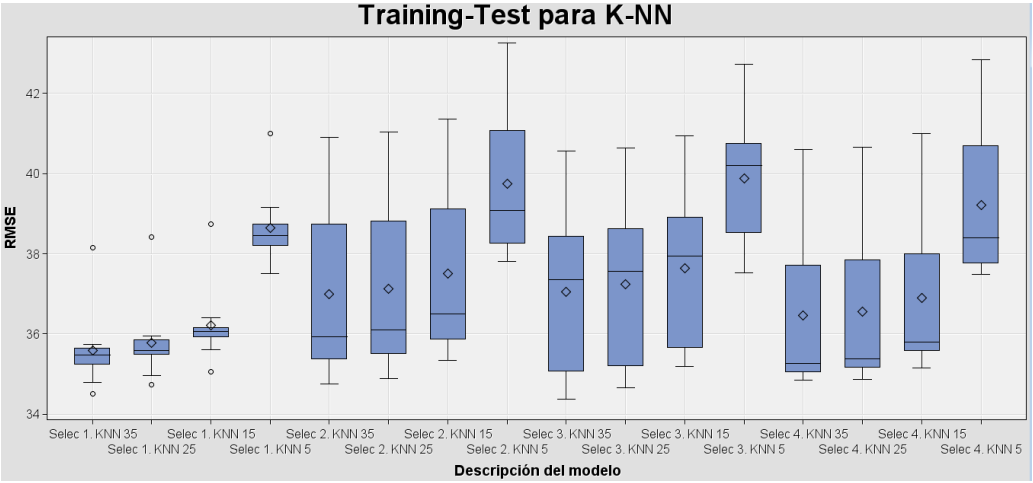


Figura 6. Diagrama de caja del training-test para k-NN

En este sentido, el mejor modelo es el segundo (**25 vecinos**), que ofrece menor variabilidad que el resto. El mejor modelo (k=25 sin variables transformadas), que se usará para comparar con los obtenidos mediante las demás técnicas, ofrece un RMSE de **35.75**, lo que se traduce en R^2 del **29%.**

5.6.2. Regresión lineal

5.6.2.1 Regresión lineal con SAS Enterprise Miner

Para la regresión lineal con SAS Enterprise Miner, se decide probar con las 4 selecciones de variables anteriormente probados para k-NN con el fin de discernir si el hecho de aplicar las técnicas de transformación o clusterización mejoran la predicción de la variable objetivo. Al volver a aplicar un training-test repetido, se podrá emitir un juicio que justifique el uso de determinada selección frente a otra.

Así mismo, se aplicarán los métodos de selección clásicos de los modelos de regresión lineal que se han comentado en el apartado de metodología. Para que la aleatoriedad no influya en los resultados, se realiza un training-test repetido con 10 iteraciones con semilla aleatoria sobre los datos introducidos a la fase de modelización los diferentes caminos a seguir.

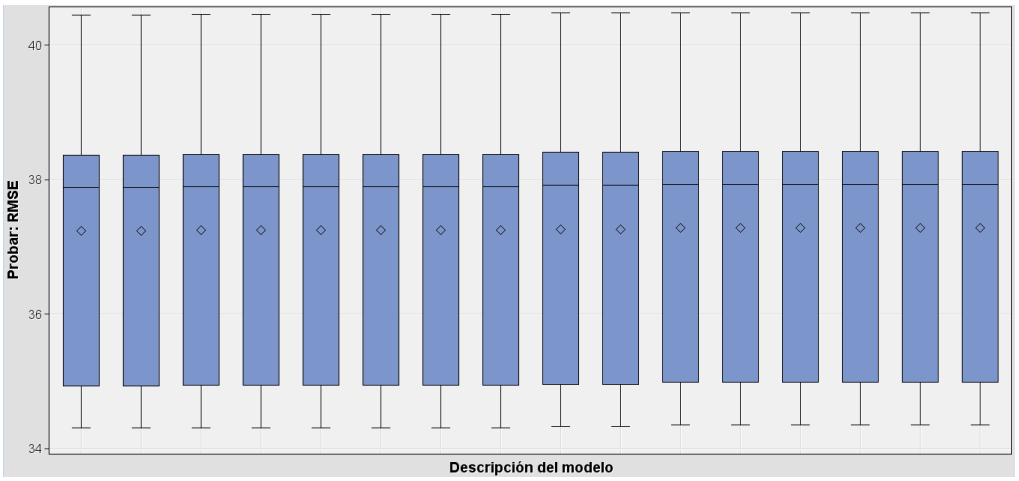


Figura 7. Diagrama de caja del training-test para la regresión lineal en SAS Miner

Como se puede observar en la Figura 7, no hay diferencia palpable entre escoger un camino u otro. Si esto lo sumamos a los resultados de k-NN, parece claro que el hecho de aplicar diferentes técnicas para intentar mejorar la predicción o bien no surten ningún efecto, o bien la empeoran, por lo que a modo general se escogerá el primer camino (archivo original con selección manual) para aplicar las diferentes técnicas de predicción, salvo diferentes modificaciones que se irán indicando en cada método, que será modelar con el archivo con variables transformadas en aquellos métodos que tengan selección de variables incorporadas para no perder información potencialmente valiosa.

En este caso en concreto, para la repetición training-test de SAS Miner, se puede observar que todas las predicciones son prácticamente iguales y con mucha variabilidad en el ASE.

5.6.2.1 Regresión lineal con R

Usar R viene dado por el hecho de que es posible usar validación cruzada repetida. Además, en este software se incluirán interacciones entre los factores del modelo, lo cual consiste en la combinación a través del producto de variables del modelo. Además, se compararán los modelos a través de criterios de selección de variables que consideran el ajuste y el exceso de parámetros como AIC y BIC. Debido a todo esto, unido a que se siguen pudiendo usar los métodos de selección clásicos para la regresión se ha creado un archivo en el que se ha aplicado transformación de variables a todas las variables de intervalo siguiendo el método de correlación máxima, para comprobar si puede existir alguna mejora en la predicción derivada a la transformación de las variables. En este archivo se dejarán tanto las variables transformadas como las originales.

Adicionalmente, se realizará una regresión LASSO. La validación cruzada repetida se realiza sobre los siguientes modelos, recogidos en la Tabla 2, con el objetivo de encontrar alguna predicción que mejore a SAS Enterprise Miner.

Tabla 2. Modelos para la Regresión lineal en R

Tipo de modelo	**Nombre del modelo**
Entrada de variables sin método de selección	Modelo 1
Regresión Stepwise con AIC	Modelo 2
Regresión Backward con AIC	Modelo 3
Regresión Stepwise con BIC	Modelo 4
Regresión Backward con BIC	Modelo 5
Regresión Stepwise con Interacciones y AIC	Modelo 6
Regresión Stepwise con Interacciones y BIC	Modelo 7
Regresión LASSO	LASSO

Los resultados de la validación cruzada repetida en términos de RMSE son los que se pueden observar en la Figura 8. Vemos que todos los modelos tienen unas medidas de error bastante similares, con atípicos cercanos o sobrepasando los 40 minutos de RMSE.

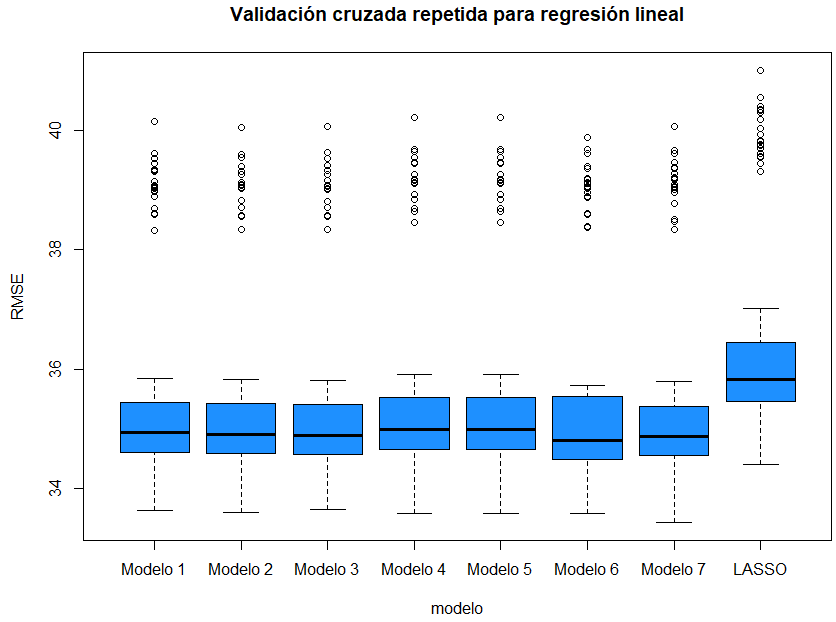


Figura 8. RMSE para la regresión lineal

Se hace por tanto necesario un segundo gráfico que muestre el R^2 junto con los parámetros de cada variable, para poder decidir sobre la sencillez o complejidad de cada uno. A igual

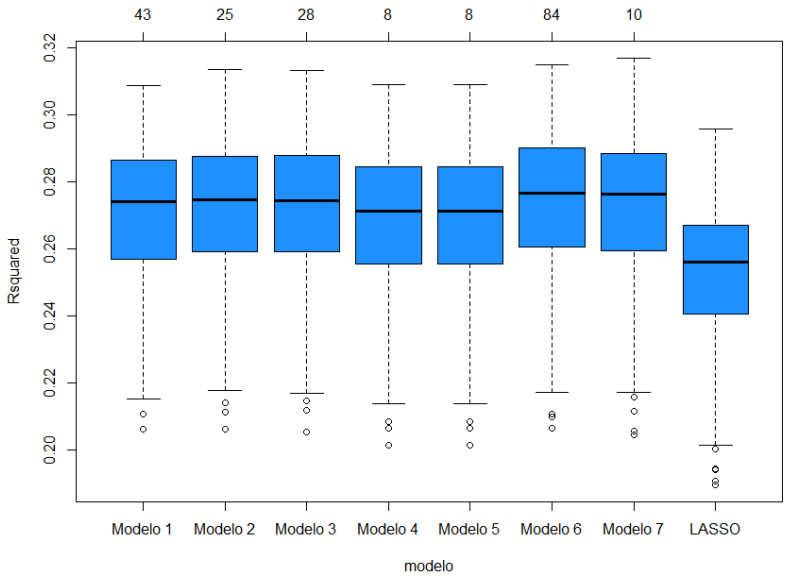


Figura 9. R-cuadrado regresión lineal en R

En la Figura 9 se encuentra la medida del R^2 de los diferentes modelos que se han obtenido junto con los parámetros de cada uno de ellos (encima del bigote superior de las cajas en la Figura 8). A excepción del LASSO, que presenta un R^2,

todos presentan unas medidas similares, aunque los modelos 4,5 y 7 tienen menos parámetros que los demás. Ya que no está muy clara la selección del mejor modelo, se pueden calcular los valores del R^2 en media y en desviación para ayudar a decidir:

Media:

	modelo	Rsquared
1	Modelo 1	0.2694843
2	Modelo 2	0.2704770
3	Modelo 3	0.2704346
4	Modelo 4	0.2669459
5	Modelo 5	0.2669459
6	Modelo 6	0.2724217
7	Modelo 7	0.2708082
8	LASSO	0.2516669

Desviación típica:

	modelo	Rsquared
1	Modelo 1	0.02236746
2	Modelo 2	0.02258022
3	Modelo 3	0.02262925
4	Modelo 4	0.02328155
5	Modelo 5	0.02328155
6	Modelo 6	0.02395455
7	Modelo 7	0.02388692
8	LASSO	0.02308812

Vemos que los modelos sobre los que se dudaba, que son Step con BIC (4), Backward con BIC (modelo 5) y Step con BIC con interacciones tienen medidas muy similares de ajuste. En concreto el 4 y el 5 son iguales, y el 7 presenta un R^2 un poco mayor. Aún así, un R^2 del 26.9% (modelos 4 y 5) y otro del 27.02% se prefiere elegir cualquiera de los dos primeros dado que tienen 2 parámetros menos, como se observa en la Figura 8. Se elige como ganador el modelo 4 (Stepwise con BIC) cuya fórmula es la siguiente:

```
lm(formula = minutes ~ best_of + G_tourney_name + min_rank_point +
    LOG_IMP_REP_min_rank, data = data_train)
```

y los valores de sus coeficientes:

```
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        85.863071   0.912725  94.073  < 2e-16 ***
best_of5           63.220850   2.983691  21.189  < 2e-16 ***
G_tourney_name1     6.191137   0.757503   8.173 3.30e-16 ***
G_tourney_name2     4.924052   2.188557   2.250   0.0245 *
G_tourney_name3   -16.425286   3.435876  -4.781 1.77e-06 ***
G_tourney_name4    -7.367172   3.159204  -2.332   0.0197 *
min_rank_points     0.005302   0.000467  11.353  < 2e-16 ***
LOG_IMP_REP_min_rank 17.822214   2.881995   6.184 6.45e-10 ***
```

Siendo la variable más importante *best_of,* el modelo guarda lógica estimando que los partidos al mejor de 5 sets son de media 63 minutos más largos que al mejor de 3 sets. Por otro lado, los partidos que más parecen durar son aquellos de los torneos que se encuentran en la categoría 1 de *G_tourney_name* (ver Tabla 1 del anexo). Con respecto a la variable *min_rank_points,* esta no parece tener mucho peso en el aumento de la duración (5 minutos por cada 1000 puntos del ranking ATP, lo cual es una cantidad bastante alta al alcance de pocos jugadores). Con respecto al ranking (*min_rank)*, el incremento del 1% en esta variable (del jugador peor clasificado) hace que la duración de los partidos aumente en 0,178 minutos más.

5.6.3 Redes neuronales

Para modelizar con redes neuronales, se sigue con el archivo que contiene las variables transformadas junto con las originales. De acuerdo con Portela (2020), es esencial seleccionar correctamente las variables que entran a formar parte de la red, pues un exceso de variables puede llevar al sobreajuste, mientras que, si se introducen menos variables de las necesarias, se puede caer en un ajuste insuficiente, lo que llevaría a aumentar el sesgo.

Se decide usar la macro **%randomselect** (Portela, 2020), que consiste en realizar un proceso *stepwise* repetido, sorteando cada vez un % de los datos *train.* La salida incluye una tabla con las frecuencias repetidas para cada combinación de variables. Para mayor fiabilidad, se ejecuta tres veces la macro cambiando el criterio de selección (glmselect, BIC y AIC). En la Tabla 3 del Anexo A.1 se puede ver la salida de la macro con los modelos que más se han repetido cada vez. Las variables contenidas en estos modelos son las que más se repiten, y que por lo tanto entran a la fase de modelización para las redes neuronales son:

- tourney_level
- best_of
- round
- min_rank_points
- G_tourney_name
- IMP_REP_min_rank
- LOG_IMP_REP_min_rank
- IMP_REP_min_age

Para la construcción de redes neuronales, se hará uso de la librería *caret* de R, en la cual existen varios paquetes que permiten crear redes. En concreto, se usarán los paquetes "nnet" y "avnnet". Ambos métodos son similares, excepto por el hecho de que el segundo sortea varias repeticiones de *training* de la red con diferentes semillas y saca el error promedio mientras que "nnet" realiza esto una sola vez. El inconveniente de usar solo "nnet" es que, si cambian los pesos de la red, la predicción se ve afectada. Por lo tanto, es conveniente usar "avvnet". Aun así, se usarán los dos métodos y se compararán los resultados obtenidos.

Como ya se ha visto, uno de los parámetros que se deben configurar en las redes es el número de nodos de la capa oculta, que dependen tanto del número de datos como de la complejidad de estos. Siguiendo los ejemplos que hay en la literatura, se puede calcular el número de nodos aproximado, que viene dado por el hecho de que deben contener al menos 20 observaciones por parámetro. Este número se puede calcular por la siguiente fórmula:

$$h(k+1)+h+1 = nº\ observaciones/20$$

Siendo h el número de nodos ocultos y k el número de nodos input. Por tanto:

$$35h + 1 = 15121/20;\ h{\sim}21\ nodos$$

No obstante, dada la incertidumbre que rodea los datos que son objeto de estudio, se probará con un número de diferente de nodos con el objetivo de intentar encontrar el mejor modelo de redes neuronales, que cobra especial importancia en este caso dado que los paquetes "nnet" y "avnnet" tienen una única capa oculta de entrada y salida. Portela (2020) indica que el número mínimo de nodos en la única capa oculta de esto paquetes debería ser 3. Por lo tanto, se irán subiendo desde este número hasta llegar al máximo recomendado para los datos de estudio, que son 21. En definitiva, se probarán modelos de redes con 3,5,10,15,18 y 21 nodos. Como *decay* se dejan los valores de 0.01,0.1 y 0.001. Aplicando estas funciones al método "nnet", se puede observar en la Figura 10, después de haberse probado todas las combinaciones posibles entre *size* y *decay* con validación cruzada repetida, que todas las redes tienen un error medio absoluto (*MAE*) muy similar, y lo mismo ocurre con el r-cuadrado (*Rsquared*), siendo las redes con 5 nodos las que parecen presentar un mejor resultado.

size	decay	RMSE	Rsquared	MAE
3	0.001	36.64594	0.2570518	28.02069
3	0.010	36.55343	0.2584710	27.95426
3	0.100	36.52376	0.2596438	27.92569
5	0.001	36.46989	0.2617287	27.86883
5	0.010	36.45628	0.2622356	27.85494
5	0.100	36.46772	0.2617913	27.86108
10	0.001	36.48386	0.2610622	27.85632
10	0.010	36.49868	0.2605376	27.87946
10	0.100	36.49315	0.2606496	27.88705
15	0.001	36.54473	0.2587574	27.93869
15	0.010	36.53542	0.2591619	27.91709
15	0.100	36.52661	0.2594632	27.91264
18	0.001	36.52245	0.2596275	27.92973
18	0.010	36.52869	0.2593457	27.92682
18	0.100	36.51688	0.2598097	27.92278
21	0.001	36.53228	0.2593016	27.91688
21	0.010	36.56811	0.2578756	27.94052
21	0.100	36.54218	0.2588739	27.93022

Figura 10. Resultados para nnet

size	decay	RMSE	Rsquared	MAE
3	0.001	36.64594	0.2570518	28.02069
3	0.010	36.55343	0.2584710	27.95426
3	0.100	36.52376	0.2596438	27.92569
5	0.001	36.46989	0.2617287	27.86883
5	0.010	36.45628	0.2622356	27.85494
5	0.100	36.46772	0.2617913	27.86108
10	0.001	36.48386	0.2610622	27.85632
10	0.010	36.49868	0.2605376	27.87946
10	0.100	36.49315	0.2606496	27.88705
15	0.001	36.54473	0.2587574	27.93869
15	0.010	36.53542	0.2591619	27.91709
15	0.100	36.52661	0.2594632	27.91264
18	0.001	36.52245	0.2596275	27.92973
18	0.010	36.52869	0.2593457	27.92682
18	0.100	36.51688	0.2598097	27.92278
21	0.001	36.53228	0.2593016	27.91688
21	0.010	36.56811	0.2578756	27.94052
21	0.100	36.54218	0.2588739	27.93022

Figura 11. Resultados para avvnet

Después, se procede con el entrenamiento de la red con el método "avnnet" con validación cruzada repetida. Se usan los mismos valores para el *size* y el *decay.*

En la Figura 11 se muestran los resultados obtenidos, que no muestran apenas diferencias con los recogidos en "nnet", si bien las redes con 5 y 10 nodos siguen pareciendo algo mejores que el resto, con un R^2 del **26%**

Se pueden comparar los mejores modelos conseguidos tanto en "nnet" y "avvnet" en un diagrama de cajas. En la Figura 12 se muestra la comparativa entre los mejores modelos que se han obtenido en validación cruzada para cada uno de los métodos.

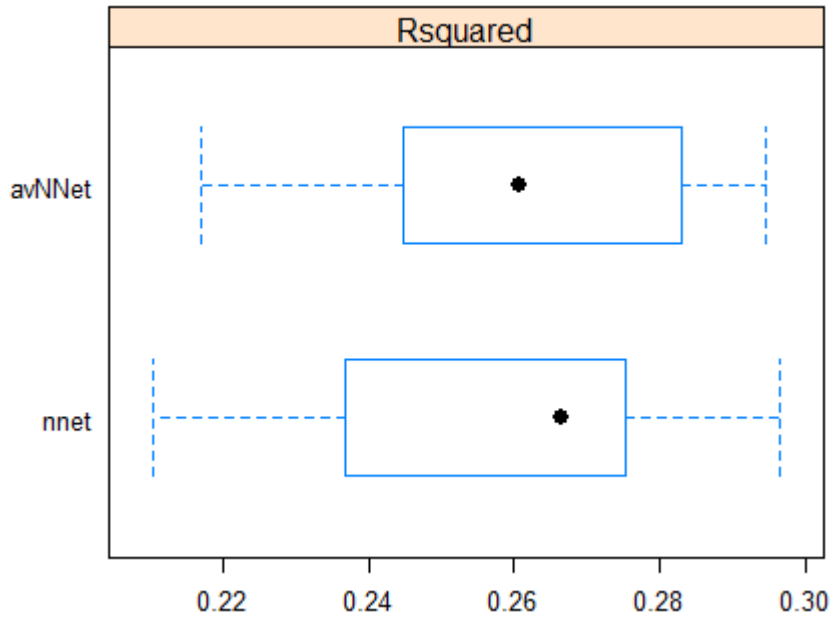


Figura 12: Diagrama de cajas comparativo entre nnet y avnnet

Se puede observar que el modelo con "avnnet" es algo mejor que el que de "nnet". Es significativo mencionar que no se consiguen mejores resultados con las redes que con el método de regresión, más clásico. Los métodos que se han probado en R usan como función de optimización "BackPropagation" y como función de activación "Tangente Hiperbólica".

Se ha probado en SAS base con más modelos de redes neuronales, en los que además de variar el número de nodos, se ha hecho estudio de Early Stopping y se han cambiado los algoritmos de optimización y las funciones de activación, sin conseguir mejores resultados que los que ya se han mostrado. Un ejemplo de esto se puede ver en el diagrama de cajas de la Figura 13, donde en el eje Y se representa el ASE y en el eje X se muestran los modelos probados en validación cruzada repetida, siendo regresiones lineales los modelos del 1 al 4 y todos los demás diferentes modelos de redes neuronales. Como se puede comprobar, las redes neuronales en SAS tampoco consiguen mejorar los resultados que hasta ahora ofrece la regresión lineal.

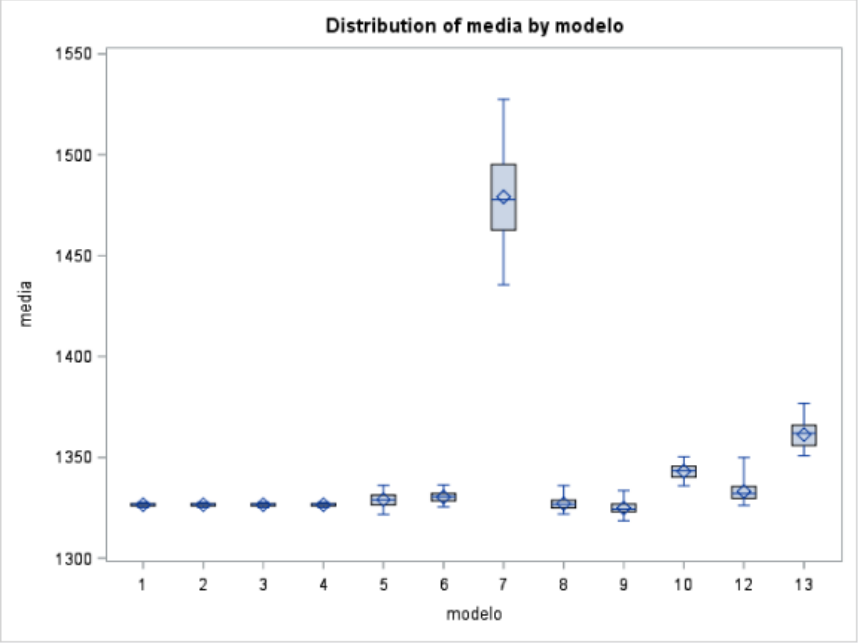


Figura 13. Diagrama de cajas para las redes neuronales en SAS

En el anexo se encuentra más detallado el proceso llevado a cabo en SAS, así como el código utilizado para la macro %randomselect y para la ejecución de las redes neuronales, en las que también se han utilizados varias macros.

5.6.4 Random Forest

Para el tuneado de Random Forest, se hará un primer estudio de la evolución de fijar o no ciertos parámetros en el algoritmo. Así, se prueba primero con el número de variables que podría ser idóneo para cada árbol. Se prueba con un *mtry* de 3,4,5,6,7,8,9,10,11,12 y 13 variables. También se prueba con un número alto de árboles, de modo que luego se estudie si se puede disminuir este parámetro. Asi, se prueba con 1000 árboles, además de un número mínimo de 10 observaciones por nodo. En la Figura 14 se observa que el mejor modelo obtenido en esta primera prueba usa 4 variables, y obtiene un R^2 del **25.86%**, aproximadamente.

mtry	RMSE	Rsquared	MAE
3	36.57717	0.2585319	27.96436
4	36.54360	0.2586610	27.93537
5	36.60774	0.2559336	27.97471
6	36.72514	0.2516488	28.05762
7	36.80801	0.2490015	28.12045
8	36.92107	0.2451998	28.20266
9	37.01078	0.2423082	28.26978
10	37.05550	0.2410791	28.28885
11	37.10927	0.2393248	28.32664
12	37.13709	0.2386662	28.34817
13	37.16451	0.2377938	28.36927

Figura 14: Resultados para Random Forest

Ahora bien, es interesante hacer un estudio del *Early Stopping*, para intentar ajustar al máximo el número de iteraciones, ya que muchos árboles añaden

complejidad al modelo. Por lo tanto, se aumenta el número de árboles a 3000 para poder decidir con suficiente margen el criterio de parada, y se fija el *mtry* en 4.

Como se puede observar en la Figura 15, la curva de error se aplana antes de las 500 iteraciones por lo que con este número de árboles o alguno menor debería ser suficiente.

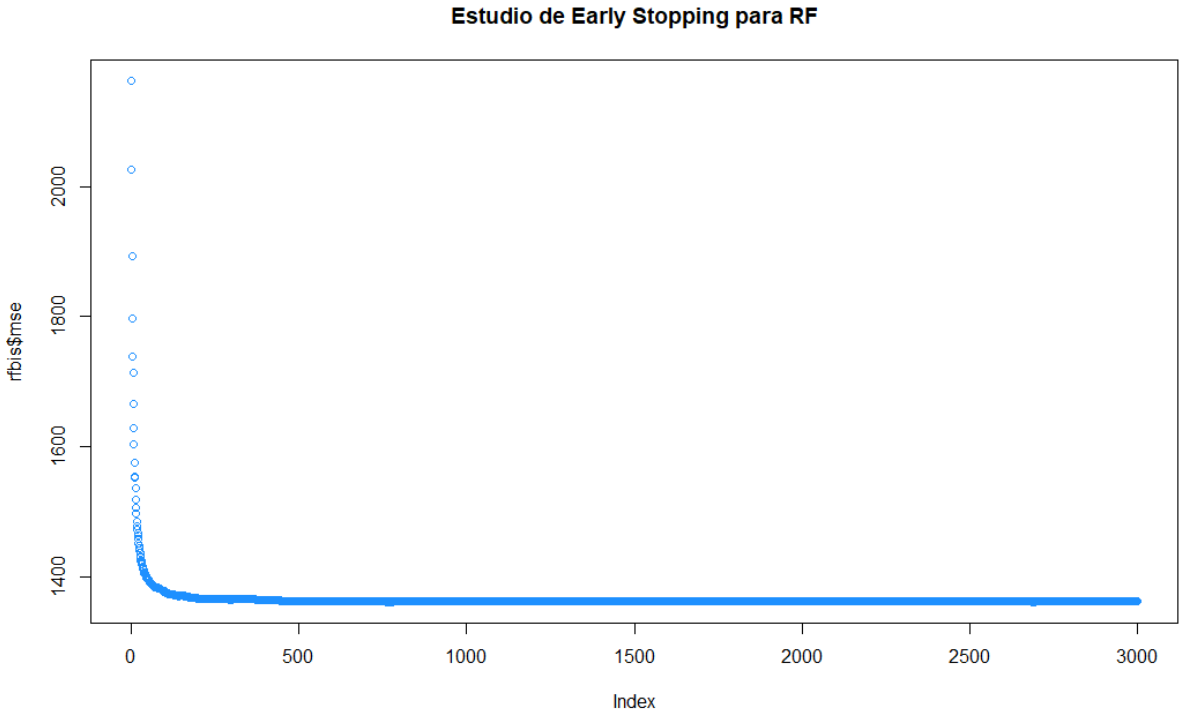


Figura 15. Estudio de Early Stopping para Random Forest.

Los modelos finales que se prueban en validación cruzada repetida son los que se pueden comprobar en la Tabla 3.

Tabla 3. Resumen de los modelos de Random Forest a comparar

Variables a sortear	Número de árboles	Tamaño mínimo de hoja	Nombre del modelo
4	500	10	RF1
4	500	30	RF2
4	400	40	RF3
4	200	50	RF4
4	100	60	RF5

Estos modelos que se han descrito se han realizado con validación cruzada repetida. Los resultados se pueden comprobar en la Figura 16. Como se puede observar, no hay mucha diferencia de error entre todos los modelos, ya que la escala del RMSE está muy acotada. Esto quiere decir que todos los modelos son prácticamente iguales. Ya que el cuarto modelo (RF4) es más sencillo y estable que el resto, se elige para evaluarlo junto con los demás algoritmos. Este modelo tiene 200 iteraciones, 50 observaciones por hoja y 4 variables a sortear en cada nodo. La medida del RMSE, 36,50 minutos, equivale aproximadamente a un R^2 del **26%.**

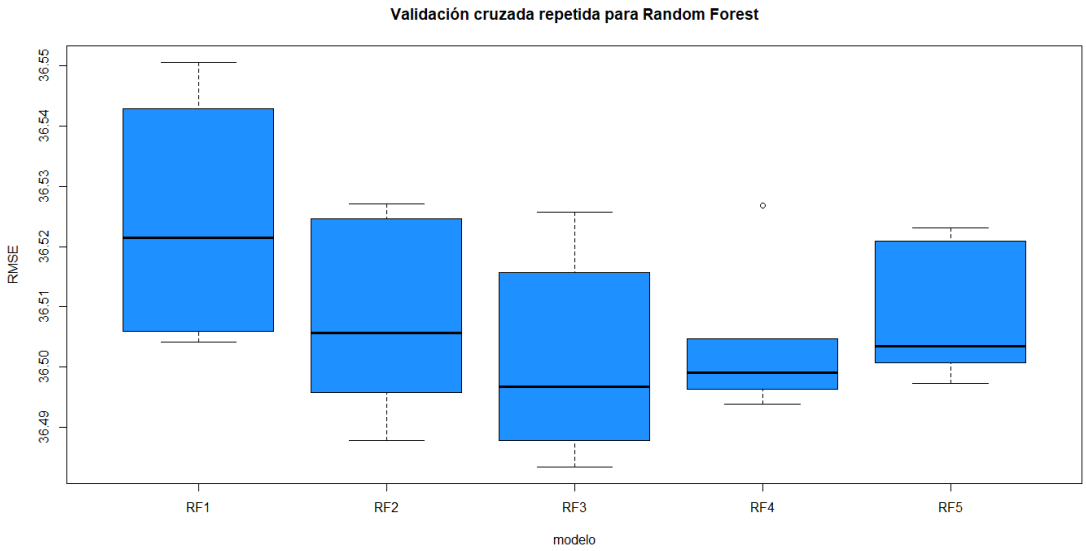


Figura 16. Resultados de validación cruzada repetida para Random Forest.

Si se estudia la importancia de variables, nos queda el gráfico de la Figura 17, en el que se ve que la variable que más importancia tiene en el modelo seleccionado es *best_of,* así como *G_tourney_name*, *tourney_level,* y *min_rank_points*. Es significativo mencionar que el modelo de Random Forest usa más variables que la regresión lineal obteniendo así mejores resultados, al ser este un modelo más estable y con menos valores atípicos que el de la regresión lineal.

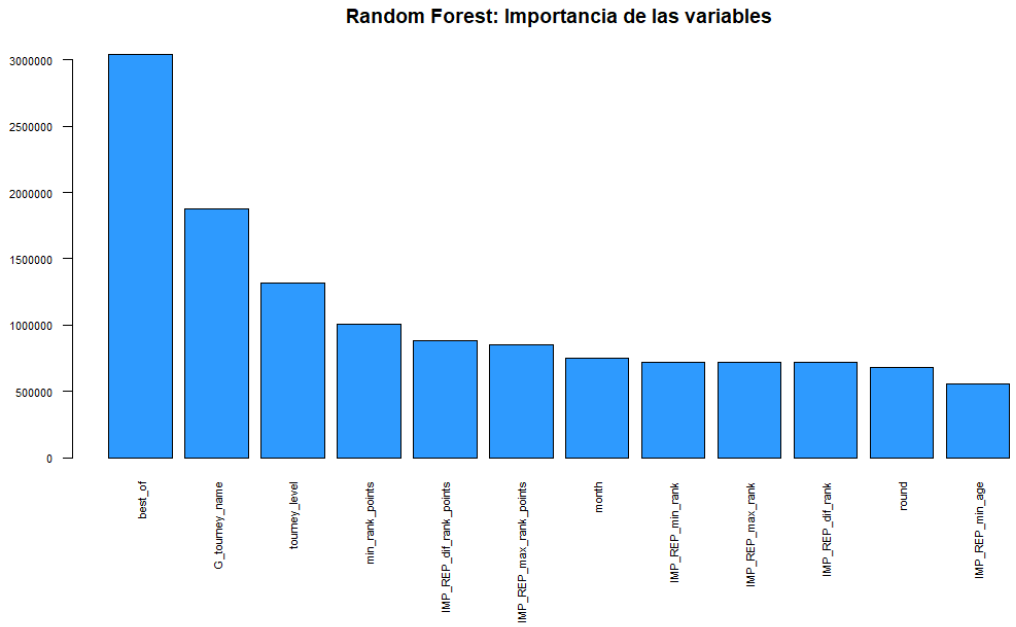


Figura 17. Importancia de las variables en Random Forest

5.6.4. Gradient Boosting

Para la configuración de Gradient Boosting se ha probado con diferentes velocidades de ajuste o *shrinkage* (0.1, 0.05, 0.03, 0.01 y 0.001), tamaños mínimos de hoja (10, 20, 40 o 60) y árboles que van desde tamaños pequeños a lo suficientemente grandes (100, 500, 1000 o 5000) para probar diferentes complejidades y dar con una buena combinación en validación cruzada. Como regla general, se ha dejado fijo el parámetro que mide la profundidad del árbol, *interaction.depth*, en 2.

shrinkage	n.minobsinnode	n.trees	RMSE	Rsquared	MAE
0.001	10	100	41.44755	0.2493508	31.36949
0.001	10	500	38.91920	0.2520434	29.55509
0.001	10	1000	37.53336	0.2526657	28.63317
0.001	10	5000	36.49749	0.2609584	27.92278
0.001	20	100	41.44755	0.2493508	31.36949
0.001	20	500	38.91920	0.2520434	29.55509
0.001	20	1000	37.53336	0.2526657	28.63317
0.001	20	5000	36.49514	0.2610778	27.91903
0.001	40	100	41.44755	0.2493508	31.36949
0.001	40	500	38.91920	0.2520434	29.55509
0.001	40	1000	37.53336	0.2526657	28.63317
0.001	40	5000	36.49160	0.2612192	27.91772
0.001	60	100	41.44755	0.2493508	31.36949
0.001	60	500	38.91920	0.2520434	29.55509
0.001	60	1000	37.53314	0.2526789	28.63301
0.001	60	5000	36.49074	0.2612476	27.91519
0.010	10	100	37.52553	0.2526914	28.62807
0.010	10	500	36.49615	0.2610011	27.92168
0.010	10	1000	36.38947	0.2648813	27.83219
0.010	10	5000	36.35882	0.2657641	27.77102
0.010	20	100	37.52553	0.2526914	28.62807
0.010	20	500	36.49307	0.2611599	27.91743
0.010	20	1000	36.37850	0.2653381	27.82582
0.010	20	5000	36.35000	0.2661306	27.76666
0.010	40	100	37.52553	0.2526914	28.62807
0.010	40	500	36.49015	0.2612601	27.91642
0.010	40	1000	36.37134	0.2656383	27.81882
0.010	40	5000	36.32955	0.2669654	27.75003
0.010	60	100	37.52529	0.2527070	28.62792
0.010	60	500	36.48982	0.2612688	27.91409
0.010	60	1000	36.36695	0.2658187	27.81411
0.010	60	5000	36.33870	0.2665849	27.75196
0.030	10	100	36.60873	0.2574278	28.00959
0.030	10	500	36.33694	0.2668291	27.77756
0.030	10	1000	36.32529	0.2670973	27.75642
0.030	10	5000	36.58486	0.2576908	27.91601
0.030	20	100	36.60726	0.2575120	28.00702
0.030	20	500	36.32392	0.2673814	27.77055
0.030	20	1000	36.31131	0.2676758	27.74810
0.030	20	5000	36.57489	0.2580602	27.91389
0.030	40	100	36.60760	0.2574948	28.00720
0.030	40	500	36.31476	0.2677533	27.76273
0.030	40	1000	36.29664	0.2682739	27.73629
0.030	40	5000	36.55259	0.2587561	27.89648
0.030	60	100	36.60557	0.2575939	28.00557
0.030	60	500	36.31054	0.2679088	27.75794
0.030	60	1000	36.30104	0.2680835	27.73149
0.030	60	5000	36.54191	0.2590941	27.88424
0.050	10	100	36.49110	0.2611540	27.91832
0.050	10	500	36.31770	0.2674469	27.75562
0.050	10	1000	36.36671	0.2654485	27.77811
0.050	10	5000	36.81145	0.2503025	28.06203
0.050	20	100	36.48748	0.2613571	27.91280
0.050	20	500	36.30747	0.2678643	27.74807
0.050	20	1000	36.35680	0.2658577	27.77412
0.050	20	5000	36.80126	0.2505862	28.06293
0.050	40	100	36.48451	0.2614091	27.91266
0.050	40	500	36.30091	0.2681333	27.74437
0.050	40	1000	36.33965	0.2665666	27.75588
0.050	40	5000	36.74712	0.2522508	28.02707
0.050	60	100	36.48593	0.2613942	27.91067
0.050	60	500	36.29399	0.2684014	27.72775
0.050	60	1000	36.34694	0.2662466	27.75460
0.050	60	5000	36.72276	0.2529456	27.99722
0.100	10	100	36.37522	0.2654604	27.81756
0.100	10	500	36.37238	0.2652274	27.78200
0.100	10	1000	36.48453	0.2611471	27.85215
0.100	10	5000	37.33084	0.2350406	28.43133
0.100	20	100	36.37323	0.2655196	27.81775
0.100	20	500	36.36375	0.2655865	27.77655
0.100	20	1000	36.48261	0.2612396	27.85822
0.100	40	5000	37.13200	0.2403645	28.29555
0.100	60	100	36.36286	0.2659518	27.80453
0.100	60	500	36.34688	0.2662513	27.75768
0.100	60	1000	36.45525	0.2621572	27.83539
0.100	60	5000	37.01546	0.2438890	28.21455

Figura 18. Resultados GBM

En la Figura 18 se muestra el resultado de todas las combinaciones, que son 80 en total, de las que se extrae que el mejor modelo (marcado en azul) habría obtenido un RMSE de 36,29 minutos, con un *shrinkage* de 0,05 y 500 iteraciones, con un tamaño de hoja mínimo de 60 observaciones. El R^2 que se obtendría con este modelo sería del 26,84%.

También se realiza una representación gráfica de los principales parámetros de Gradient Boosting mostrando los diferentes errores obtenidos. Esto se puede ver en la Figura 19, de la cual se puede deducir que a partir de unos valores similares del *shrinkage* para todas las iteraciones (en torno a 0.03 o menos), los modelos no mejoran en cuanto a error

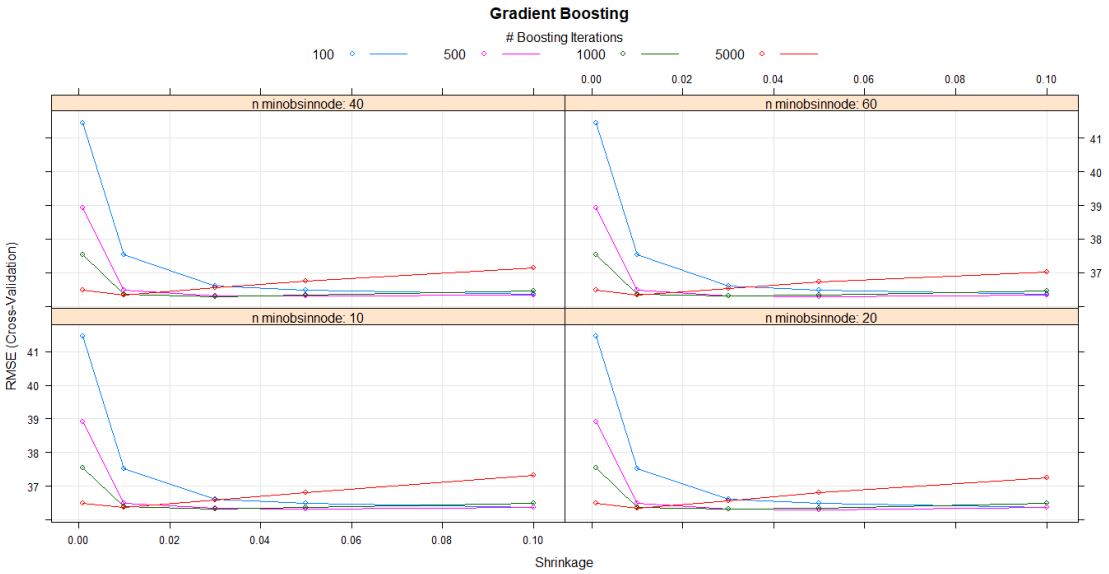


Figura 19. Representación gráfica de los principales parámetros de Gradient Boosting

Acto seguido, se puede estudiar el Early Stopping para ver si puede ser conveniente parar en un número determinado de iteraciones o de árboles.

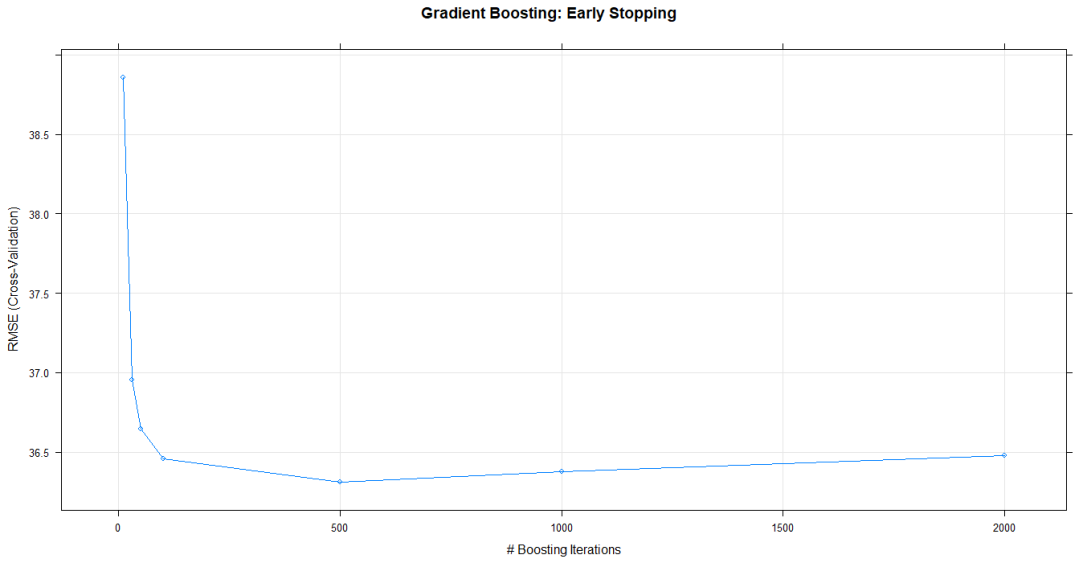


Figura 20. Estudio de Early Stopping para Gradient Boosting

El resultado del estudio se muestra en el gráfico que se dibuja en la Figura 20. El error disminuye de forma estable entre 100 y 500 iteraciones, donde alcanza un mínimo, lo que vuelve a sugerir la idea de que los datos de los que se dispone se ven perjudicados cuanto más complejo es el algoritmo. Debido a esto y en base a los resultados anteriores, se probarán varios modelos de Gradient Boosting en validación cruzada repetida. Estos modelos son los que se aprecian en la Tabla 4.

Tabla 4. Resumen de los modelos de Gradient Boosting a comparar

Número de observaciones por hoja	Número de árboles	Shrinkage	Nombre del modelo
60	500	0,03	GBM1
80	500	0,03	GBM2
80	300	0,03	GBM3
90	200	0,01	GBM4
100	100	0,01	GBM5
150	100	0,01	GBM6

Como se puede observar, se ha variado sobre todo el tamaño de hoja, aumentándolo más al tener un mayor número de observaciones, y el número de árboles, desde 500 hasta 100. Los resultados de la validación cruzada repetida son los que se pueden observar en la Figura 21.

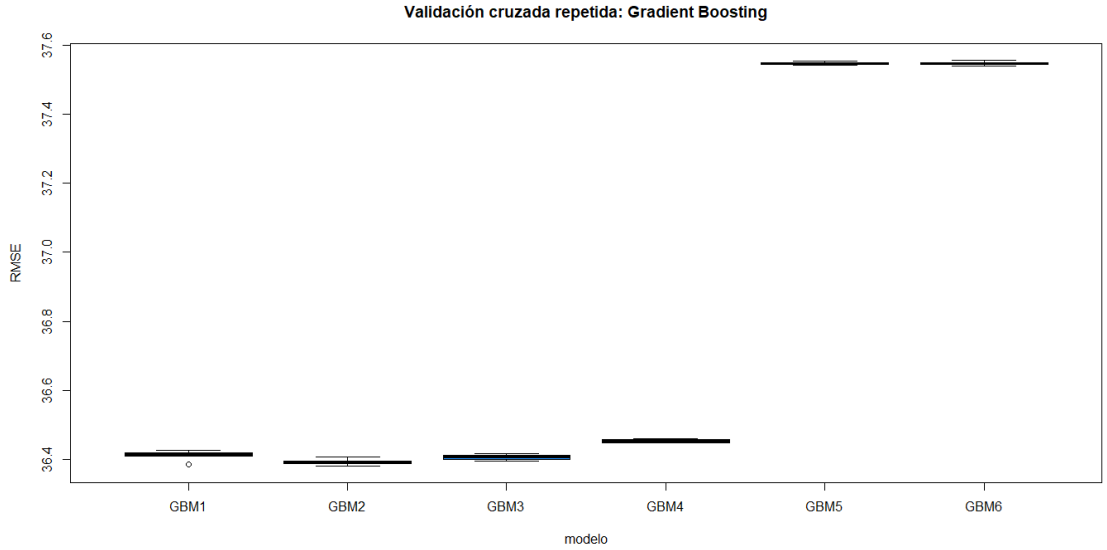


Figura 21. Resultados de validación cruzada repetida en Gradient Boosting

Se puede ver que todos los modelos son prácticamente iguales en cuanto a error, ya que giran en torno a los 36 minutos de RMSE, menos los dos últimos, que están algo más alejados. Se escoge el modelo "GMB 4" ya que, aunque tenga algo más de error que los tres primeros, la diferencia es casi imperceptible y se gana en sencillez (200 árboles). Este modelo tiene un *shrinkage* de 0.01 y 90 observaciones por hoja.

La importancia de las variables en el modelo ganador de Gradient Boosting aparece en la Figura 22. Se vuelve a comprobar que lo que más afecta a la duración de los partidos es la variable *best_of*. La variable *G_tourney_name* vuelve a ocupar el segundo lugar en la importancia, al igual que ocurría en Random Forest. Sin embargo, al contrario que en la anterior técnica, en la que la variable *tourney_level* gozaba de una importancia relativamente alta, aquí no parece ser nada importante según el algoritmo de Gradient Boosting.

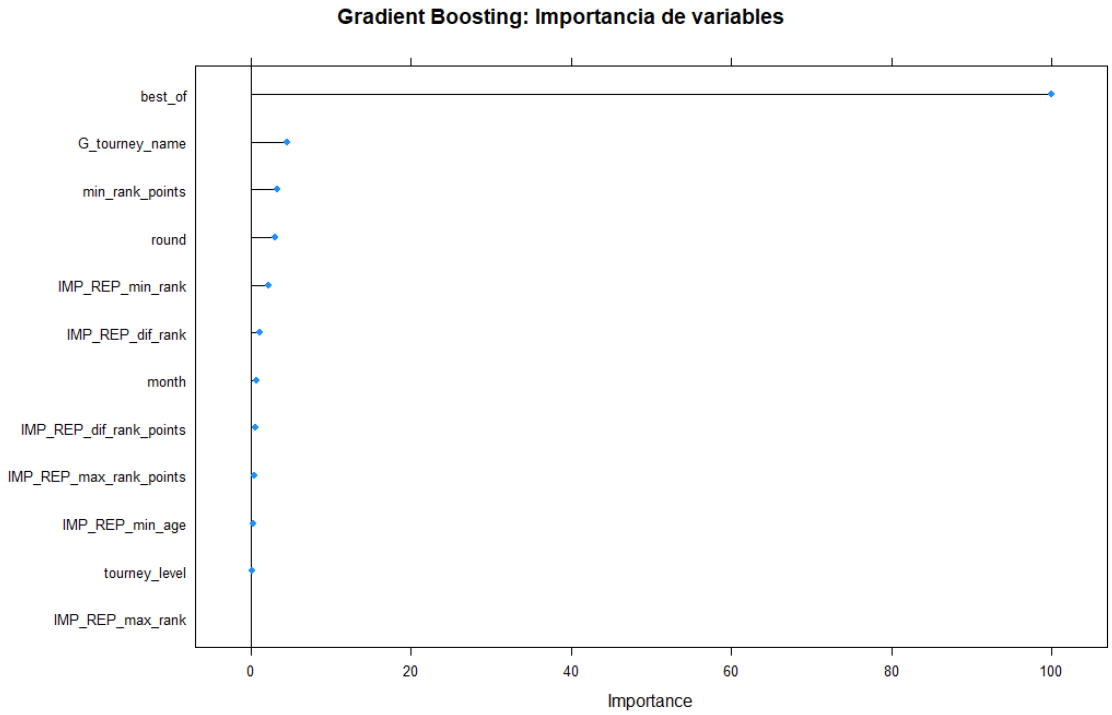


Figura 22. Importancia de variables en Gradient Boosting

5.6.5 XGBoost

Al igual que se ha hecho con Gradient Boosting, siendo XGBoost una mejora de este, se realiza un primer estudio en el que se prueba entre varios *shrinkage* (0.01, 0.03, 0.05, 0.1, 0.2 y 0.3), número de árboles o *nrounds* (100, 300, 500, 1000, 2000, 4000 y 5000). La profundidad máxima se fija en 6, al igual que el número de observaciones de las hojas finales, que serán 20, mientras que en este intento no se modifica el parámetro *gamma,* cuyo valor es 0.

Los resultados de validación cruzada en estos modelos se pueden observar en la Figura 23. De las 42 combinaciones que se prueban, el mejor modelo se ha obtenido usando 500 iteraciones, con un *shrinkage* de 0.01. El RMSE que se ha obtenido es de **36.61** minutos aproximadamente, con un R^2 del **25,61%.**

eta	nrounds	RMSE	Rsquared	MAE					
0.01	100	54.09648	0.2554013	41.26408	0.10	500	37.85765	0.2190514	28.75225
0.01	300	36.93649	0.2577319	27.52582	0.10	1000	38.66157	0.2006045	29.34482
0.01	500	36.61641	0.2561907	27.84196	0.10	2000	39.79223	0.1793351	30.17515
0.01	1000	36.72669	0.2523902	27.99012	0.10	4000	40.99232	0.1611798	31.09922
0.01	2000	36.99174	0.2437553	28.15911	0.10	5000	41.35969	0.1561562	31.38972
0.01	4000	37.49142	0.2289357	28.50075	0.20	100	37.25354	0.2355271	28.34129
0.01	5000	37.68384	0.2238100	28.62177	0.20	300	38.27807	0.2087165	29.04943
0.03	100	36.92550	0.2573015	27.53306	0.20	500	38.94217	0.1947396	29.55778
0.03	300	36.71357	0.2527563	27.97460	0.20	1000	40.11999	0.1734537	30.44243
0.03	500	36.89596	0.2466955	28.10319	0.20	2000	41.42891	0.1544886	31.48302
0.03	1000	37.29082	0.2345195	28.36342	0.20	4000	42.37898	0.1429411	32.25517
0.03	2000	37.94390	0.2166388	28.80983	0.20	5000	42.58198	0.1406384	32.41884
0.03	4000	38.76258	0.1985577	29.41536	0.30	100	37.80963	0.2204701	28.76368
0.03	5000	39.11242	0.1915244	29.67495	0.30	300	39.26903	0.1881033	29.84457
0.05	100	36.62160	0.2559703	27.85216	0.30	500	40.12680	0.1728032	30.44627
0.05	300	36.90829	0.2462793	28.11833	0.30	1000	41.36478	0.1549833	31.40502
0.05	500	37.21494	0.2366318	28.31994	0.30	2000	42.40894	0.1432430	32.25378
0.05	1000	37.75809	0.2214874	28.68518	0.30	4000	42.98673	0.1370844	32.73547
0.05	2000	38.50236	0.2039875	29.23315	0.30	5000	43.05486	0.1364206	32.78869
0.05	4000	39.54183	0.1839929	29.99336					
0.05	5000	39.92477	0.1773563	30.30116					
0.10	100	36.82654	0.2488602	28.05482					
0.10	300	37.40681	0.2313631	28.44930					

Figura 23. Resultado para XGBM

Es conveniente realizar un estudio de *Early Stopping* que nos recomiende fijar el criterio de parada en un número de iteraciones menor, y así ganar en sencillez. En la Figura 24 se puede observar el gráfico que se obtiene contraponiendo el número de iteraciones o de árboles al RMSE para cada tipo de *shrinkage* que se ha usado anteriormente. Se comprueba que, en el fondo, independientemente del parámetro de velocidad de ajuste, se alcanza un error muy similar cuantas menos iteraciones existan en el modelo. Se prefiere esto último, dado que los modelos ganan en sencillez e incluso aumentan el error a medida que se sube el número de iteraciones.

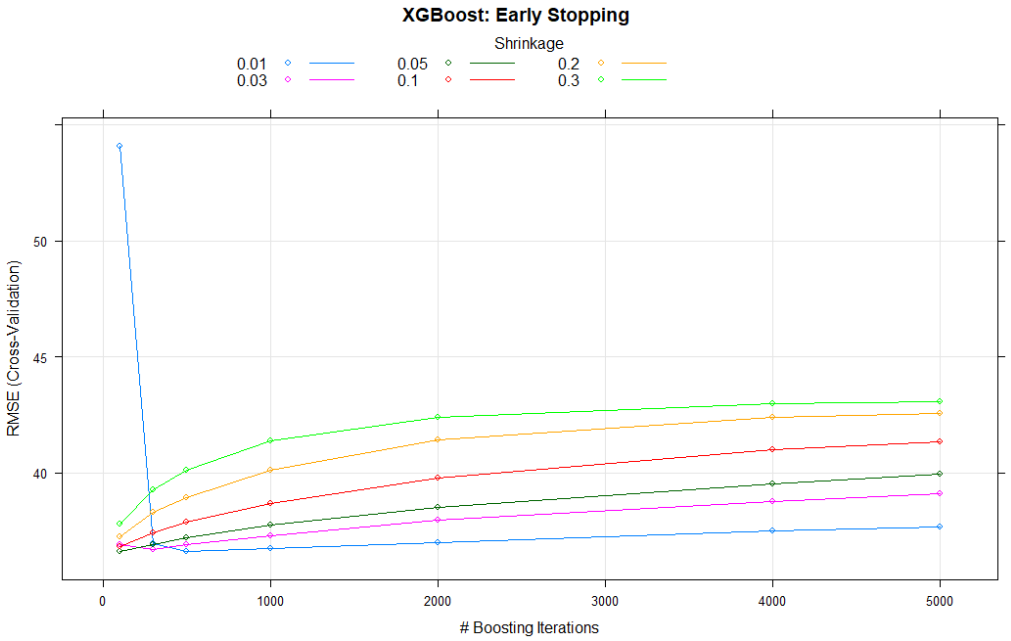


Figura 24. Early Stopping para XGBoost.

Se puede evaluar la penalización del factor *gamma* por lo que, para ello, se conservan los valores de los parámetros del modelo que ha resultado ganador en la prueba anterior, y se prueban con varios valores de *gamma* entre 0 y 1. En la Figura 25 se observan los resultados después de esta ejecución, en los que se comprueba que los cambios en el factor de regularización *gamma* no afectan en absoluto a los resultados pues se obtienen los mismos modelos.

gamma	RMSE	Rsquared	MAE
0.000	36.56925	0.2574481	27.87928
0.001	36.56925	0.2574481	27.87928
0.010	36.56925	0.2574481	27.87928
0.050	36.56925	0.2574481	27.87928
0.100	36.56925	0.2574481	27.87928
0.500	36.56925	0.2574481	27.87928
1.000	36.56925	0.2574481	27.87928

Figura 25. Resultados de XGMB

Finalmente, se realiza validación cruzada repetida de los modelos que aparecen en la Tabla 5. Se cambia el tamaño de hoja, aumentando progresivamente desde el primer modelo, al tener un número grande de observaciones, y se prueba con los valores de *shrinkage* que menor error han obtenido en las combinaciones anteriores.

Tabla 5. Modelos de XGBoost a comparar

Shrinkage	Observaciones por hoja	Número de árboles	Nombre del modelo
0,01	20	500	XGBM1
0,01	40	500	XGBM2
0,03	50	300	XGBM3
0,03	70	200	XGBM4
0,01	80	200	XGBM5
0,03	90	100	XGBM6

Los resultados de la ejecución son los observables en la Figura 26. Se puede observar que sigue sin haber prácticamente ninguna diferencia con estos métodos que la que se ha obtenido en otros algoritmos, dando lugar, en casi su totalidad, a modelos muy parecidos unos de otros. Por ser algo más estable que el resto y tener menos error, se escoge el modelo "XGBM4" como ganador de XGBoost y para la fase de evaluación con otros modelos.

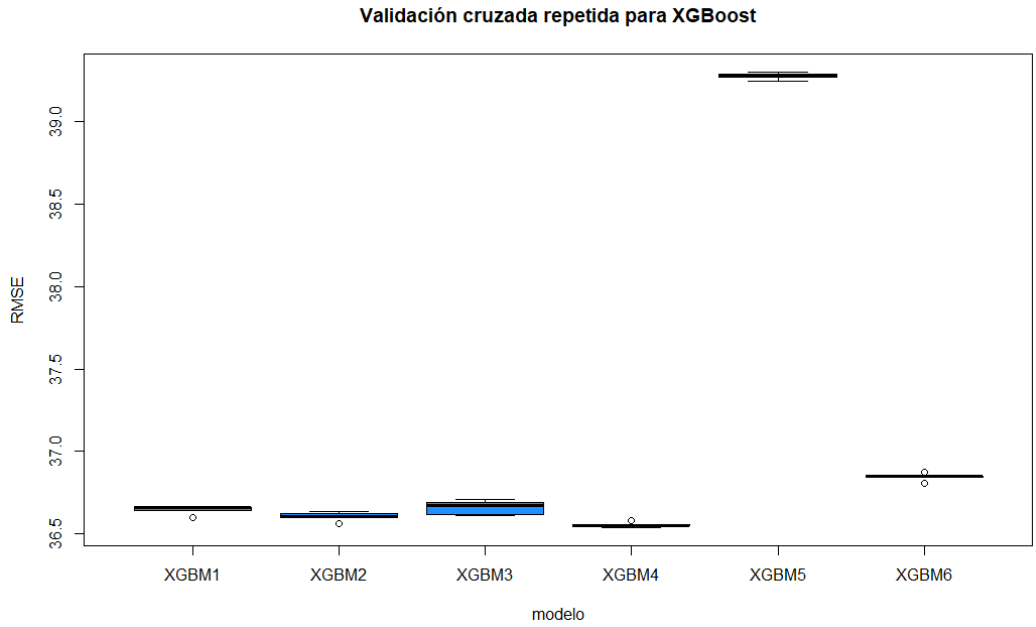


Figura 26. Resultados de la validación cruzada repetida para XGBoost

La importancia de variables para el algoritmo de XGBoost se puede ver en la Figura 27. Para la correcta ejecución de los modelos mediante esta técnica en R, es necesario que se conviertan las variables de clase a *dummies*, al igual que ocurría en k-NN o en redes neuronales. Por este motivo, se pueden ver resultados aparentemente extraños como tanta importancia para algunos niveles de ciertas variables y tan poca para otros, como puede ser el caso de *best_of*. La variable de intervalo que parece cobrar más importancia mediante este algoritmo es la diferencia entre los puntos del ranking ATP (*IMP_REP_dif_rank_points)*, que, si se fija la vista en el gráfico de importancia para Gradient Boosting (Figura 22), tenía menos importancia relativa que otras variables.

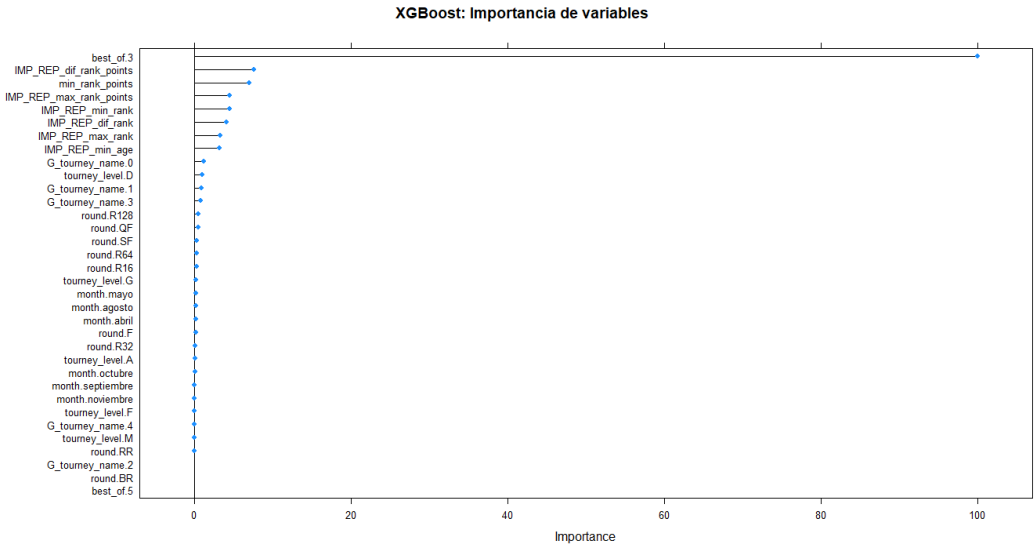


Figura 27. Importancia de variables para XGBoost

5.6.6. Comparación de modelos

Una vez se han configurado y probado los modelos, se realiza validación cruzada con 20 repeticiones para determinar qué modelo es mejor para la predicción de los minutos de un partido de tenis. Los modelos que aparecen en la Figura 28 son los seleccionados como ganadores en los apartados anteriores para cada técnica.

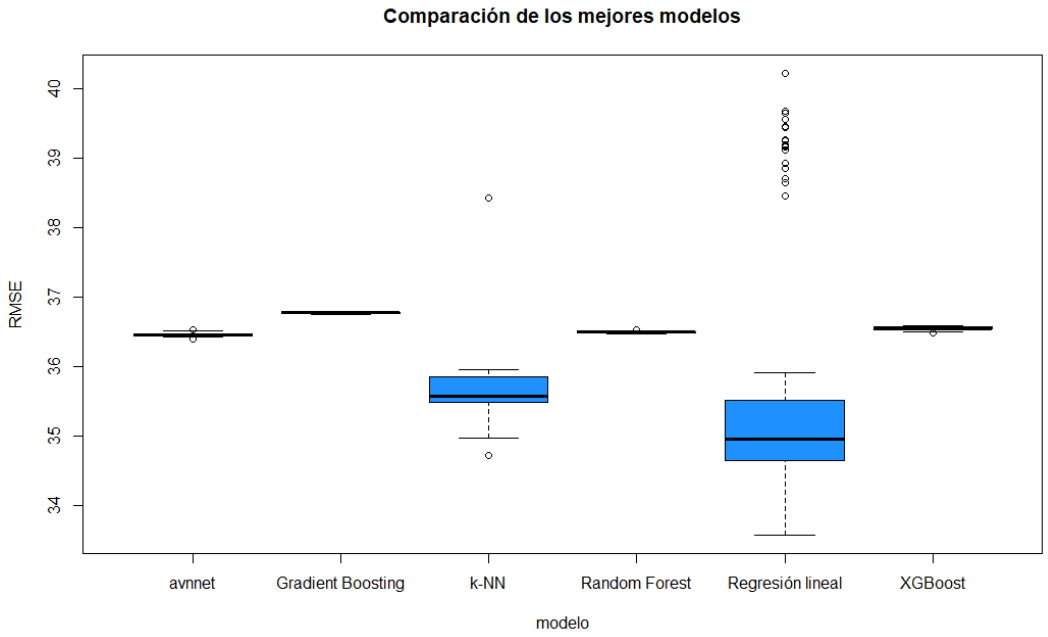


Figura 28. Comparación final de los mejores modelos para la predicción de la duración de un partido de tenis

Como se puede ver en el diagrama de cajas de la Figura 28, la regresión lineal, pese a ser un algoritmo sencillo y parecer tener un error más bajo que el resto, es más inestable, y tiene demasiados valores atípicos en la parte de arriba. Lo mismo

ocurre con k-NN, que suele ganar al resto de técnicas más complejas, pero la varianza y el atípico alejado del resto nos hace pensar en que no sea un buen modelo ganador. Observando el resto de los modelos, tanto Random Forest como XGBoost parecen estar empatados en cuanto a RMSE se refiere. Por ser más sencillo y poder identificar mejor la importancia de las variables, se selecciona el modelo con Random Forest como posible ganador para la predicción de la duración de un partido de tenis. Se intentará mejorar la predicción mediante técnicas de ensamblado, que se compararán con los resultados de este modelo.

5.6.7 Ensamblado de modelos

Se realiza un ensamblado de modelos con el objetivo de mejorar la predicción de los algoritmos que se han ido obteniendo anteriormente. De este modo, se realizan varias combinaciones consistentes en unir las predicciones de las anteriores técnicas (que se han guardado previamente) y dividir por el número de técnicas usadas en cada caso, para obtener la predicción media. Los modelos que se prueban y el nombre que reciben son los que se pueden ver en la Tabla 6.

Tabla 6. Modelos de ensamblado

Tipo de modelo	Nombre del modelo
Red Neuronal + Regresión	predi7
Random Forest + k-NN	predi8
Random Forest + Gradient Boosting + XGBoost	predi9
Regresión + k-NN + Random Forest	predi10
k-NN + Red Neuronal + XGBoost	predi11
Regresión + Red Neuronal + Random Forest	predi12
Gradient Boosting + Random Forest + Xgboost + k-NN	predi13

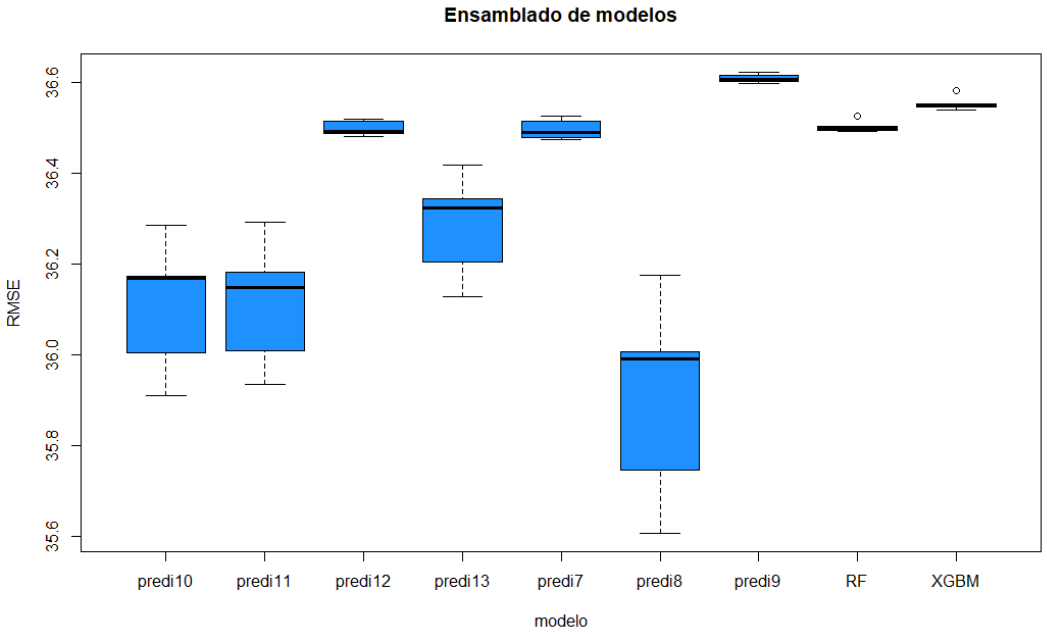


Figura 29. Ensamblado de modelos

Los resultados del ensamblado de modelo se pueden ver en la Figura 29. En este sentido, hay que mencionar que no se incluye en la comparativa los otros modelos individuales, ya que se ha visto que ofrecen peores resultados.

Tal y como se puede comprobar en la Figura 29, es cierto que los ensamblados consiguen reducir el error de los métodos que se habían seleccionado como "mejores" en la fase anterior. No obstante, es cierto que la mejora es de un 1 minuto, sobre todo en el caso de "predi8", no pareciendo merecer la pena la pérdida relativa de estabilidad que se tiene con Random Forest, siendo a su vez modelos más complicados que por separado. Por lo tanto, se escoge el modelo de Random Forest como modelo ganador para la predicción de la duración de un partido de tenis.

6. Objetivo II: Explicación y clasificación para Rafael Nadal

Para este conjunto de datos se ha extraído la información de los partidos que ha disputado Rafael Nadal entre los años 2011 y 2018, ambos inclusive.

6.1 Trabajo previo

Al igual que en el conjunto de datos de la predicción de la duración de los partidos, se ha llevado a cabo un trabajo previo en Excel que facilite su posterior exploración en SAS. En este conjunto de datos se han dejado las estadísticas individuales de cada jugador y se han diferenciado para el jugador español y para sus oponentes, para que toda la información quede en las mismas columnas y no se diferencie entre ganador y perdedor del partido.

Estas métricas de rendimiento se han incluido con los prefijos "rf" y "op" delante, donde "rf" significa que la información pertenece a Rafael Nadal y "op" que la información es de su oponente. Las variables que se mencionan se destacan en la Tabla 7 mediante un asterisco. Cabe destacar que las variables referentes a la información del torneo y la duración se siguen manteniendo para aportar más riqueza al conjunto. Se importa el archivo en SAS EM con 772 observaciones y 47 variables.

6.2. Descripción de las variables

Tabla 7. Descripción de las variables II

Variable	Descripción
tourney_name	Nombre del torneo.
surface	Tipo de superficie (Pista, tierra, hierba, alfombra o desconocida).
tourney_level	Categoría del torneo. (A=Abierto, M = Masters, G=Grand Slams, F=Finales (ATP,Olimpicas o NextGen) D=Copa Davis
month	Mes en el que se juega el torneo. Extraída de la variable fecha del dataset original.
best_of	Hace referencia al número de sets que es necesario conseguir para ganar el partido (5 o 3).
round	Ronda del torneo (octavos, cuartos, semifinales…).
minutes	Duración en minutos del partido.
hand*	Mano del tenista.
ht*	Altura en centímetros.
age*	Edad en el momento del partido.
ace*	Nº aces en el partido.
df*	Nº de dobles faltas en el partido.
svpt*	Porcentaje de servicio en el partido.
1stIn*	Porcentaje de primer servicio en el partido.
1stWon*	Porcentaje de puntos ganados en primer servicio.

2ndWon*	Porcentaje de puntos ganados en segundo servicio.
SvGms*	Números de juegos jugados en servicio.
bpSaved*	Breakpoints salvados.
bpFaced*	Breakpoints enfrentados.
rank*	ranking ATP del jugador antes del partido.
rank_points*	puntos ATP del jugador antes del partido.
win_or_lose	Variable dicotómica. 1 si Nadal gana y 0 si pierde. Es la variable objetivo.

Es importante mencionar que además se han creado variables "diferenciadoras" de cada variable de intervalo que aparece en el modelo anterior. De este modo, cada variable llevará el prefijo "dif", que indicará la diferencia entre las estadísticas de Rafael Nadal y las de su oponente. A diferencia del conjunto de datos de la predicción, estas diferencias no están en valor absoluto, sino que guardan el signo correspondiente para poder diferenciar entre Rafael Nadal y su oponente.

6.3. Exploración de datos en SAS Enterprise Miner

Después de las modificaciones en Excel, el archivo de datos no cuenta apenas con *missings*, únicamente los debidos al tratamiento de los datos atípicos. El tratamiento de atípicos y ausentes se ha hecho igual que en el primer objetivo de este trabajo, al ser la base de datos de partida la misma, con bastante información y homogeneidad.

El resultado de la exploración de las variables de clase aparece en la Figura 30.

Rol de los datos	Nombre de la variable	Rol	Número de niveles	Ausente	Moda	Porcentaje moda	Moda2	Porcentaje Moda2
TRAIN	best_of	INPUT	2	0	3	71.76	5	28.24
TRAIN	op_hand	INPUT	2	0	R	89.12	L	10.88
TRAIN	round	INPUT	9	0	R32	20.47	R16	20.21
TRAIN	surface	INPUT	3	0	Hard	54.27	Clay	38.08
TRAIN	tourney_level	INPUT	5	0	M	42.36	G	27.85
TRAIN	tourney_name	INPUT	38	0	Roland Garros	8.68	Australian Open	7.12
TRAIN	win_or_lose	TARGET	2	0	1	84.20	0	15.80

Figura 30. Resumen estadístico de las variables de clase para el conjunto de datos de clasificación

Se puede apreciar que las pistas en las que más ha jugado el tenista español durante el período de tiempo estudiado son en pista dura y en tierra batida, y que el torneo que más partidos ha jugado ha sido Roland Garros (lo cual guarda sentido si se tiene en cuenta el dato de que es el jugador de la historia que más veces lo ha ganado, lo cual implica llegar a la final).

En la Tabla 8 se muestra la información sobre el reparto de 1s y de 0s de la variable objetivo *win_or_lose*. Se puede apreciar que hay un desbalanceo en el reparto de las victorias y las derrotas, habiendo ganado Rafael Nadal casi el 85% de los 772 partidos que aparecen en la base de datos.

Tabla 8. Proporción del reparto de la variable objetivo en el conjunto de datos.

1: Rafael Nadal gana el partido	**0**: Rafael Nadal pierde el partido
84,19%	15,80%

Al igual que se hizo en el conjunto de datos de predicción de la duración de los partidos se toma la decisión de agrupar las variables de clase en niveles más sencillos, con el objetivo de crear menos categorías y no perder excesiva información de variables que tienen demasiados niveles diferentes. En el Anexo B.1 se puede ver la tabla de las nuevas variables agrupadas que se han obtenido.

6.4 Importancia de las variables

Con respecto a la importancia de las variables input con la variable objetivo, se puede usar un nodo explorador de estadísticos para determinar qué variables afectan más a que el tenista español Rafael Nadal gane o pierda un partido de tenis.

Observando la Figura 4, que contiene el valor de las variables, tal y como se ha definido en la sección anterior, se puede determinar que los puntos de break a los que tienen que hacer frente el oponente y Rafael Nadal, los puntos del ranking y el propio ranking del oponente, además de la diferencia en el ranking, son las variables que más influyen en la variable objetivo. Por otro lado, variables como el nivel del torneo (*tourney level)*, la duración de los partidos que viene representada en *minutes*, o el ranking de Rafael Nadal en el momento del partido (*rf_rank*) son un ejemplo de variables que no tienen relación en si el tenista español gana o pierde. Resulta llamativa que la superficie (*surface)*, también está incluida en las variables con poca importancia con la variable objetivo en este caso, dado la conocida maestría del tenista objetivo de este estudio en tierra batida.

6.5. Selección de variables

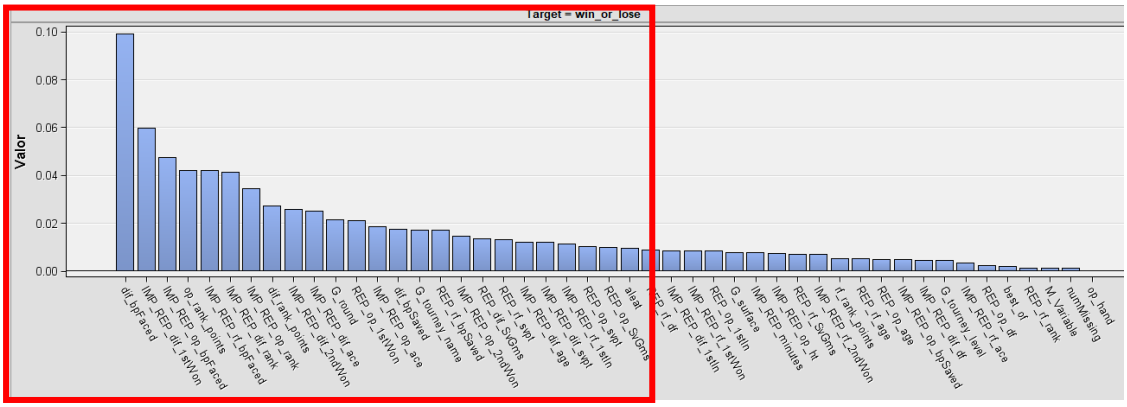


Figura 31. Selección de variables.

Al igual que en el conjunto de datos anterior, se llevará a cabo una selección inicial de variables que elimine todas aquellas variables que no influyen en la variable objetivo para que el resto de las técnicas de predicción no partan en desventaja de las que sí seleccionan automáticamente las variables más importantes. Para esto, es de nuevo de gran ayuda la creación de la variable aleatoria.

En la Figura 31 aparecen marcadas en un rectángulo todas aquellas variables que tienen mayor relación con la variable objetivo que la aleatoria. Por tanto, aquellas que estén por debajo de la misma tienen una escasa o ninguna importancia en el modelo, y serán rechazadas.

6.6. Modelización

6.6.1. Árboles de clasificación

Se llevará a cabo un training-test aleatorio con 10 repeticiones para intentar encontrar el mejor modelo de árboles de clasificación que expliquen la variable objetivo. No se prueba con selección de variables, más allá de la preliminar que se ha realizado, debido a que los árboles seleccionan las variables que mejor se adaptan al modelo, además de permitir observaciones con valores atípicos y faltantes, dada su robustez.

A continuación, en la Figura 32, se muestran los resultados del training-test repetido 10 veces. Como regla general, se ha optado por árboles binarias, es decir, con dos ramas máximas. La profundidad máxima se ha ido variando, desde 5, hasta 12 para luego aplicar poda en los árboles que se indica. Con respecto a los árboles con métodos de transformación se ha aplicado el método "Mejor" a las variables de intervalo, que maximiza el estadístico chi-cuadrado con la variable objetivo. También se ha probado con el ajuste de Bonferroni para uno de los árboles, que favorece el uso de las variables categóricas frente a las de intervalo.

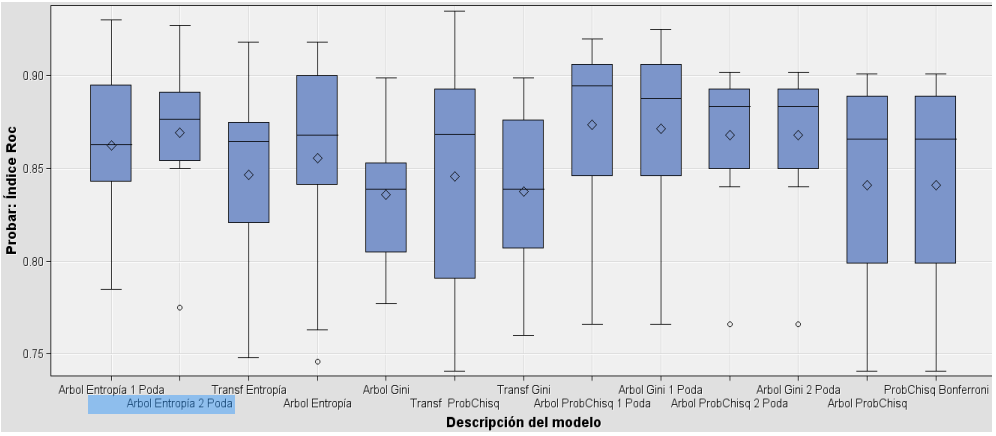


Figura 32. Diagrama de cajas para los árboles de clasificación

Dado que presentan unas medidas de clasificación errónea muy similares, nos decantaremos por el árbol que mayor fiabilidad ofrece bajo la curva ROC. El más estable es el "Árbol Entropía 2 Poda". A este árbol se le ha especificado un tamaño de hoja de 10 observaciones, y una profundidad máxima de 8 para aplicar posteriormente poda, mediante el criterio objetivo nominal de Entropía.

La importancia de las variables del modelo seleccionado se muestra en la Tabla 4, de la que podemos concluir que, según el árbol, las variables que más afectan a si Rafael Nadal gana o pierde un partido, son la diferencia de puntos de break entre los dos jugadores (*dif_bpFaced*), la que más, y la diferencia entre los primeros servicios ganados (*dif_1stWon*). Las demás variables no tienen importancia según considera el árbol, por lo que no se muestran en la tabla.

Tabla 9. Importancia de las variables del árbol.

Nombre de la variable	Número de reglas de división	Importancia
dif_bpFaced	2	1.0
dif_1stWon	2	0.5972378

Por último, en la Figura 33, se observa el diagrama final del árbol.

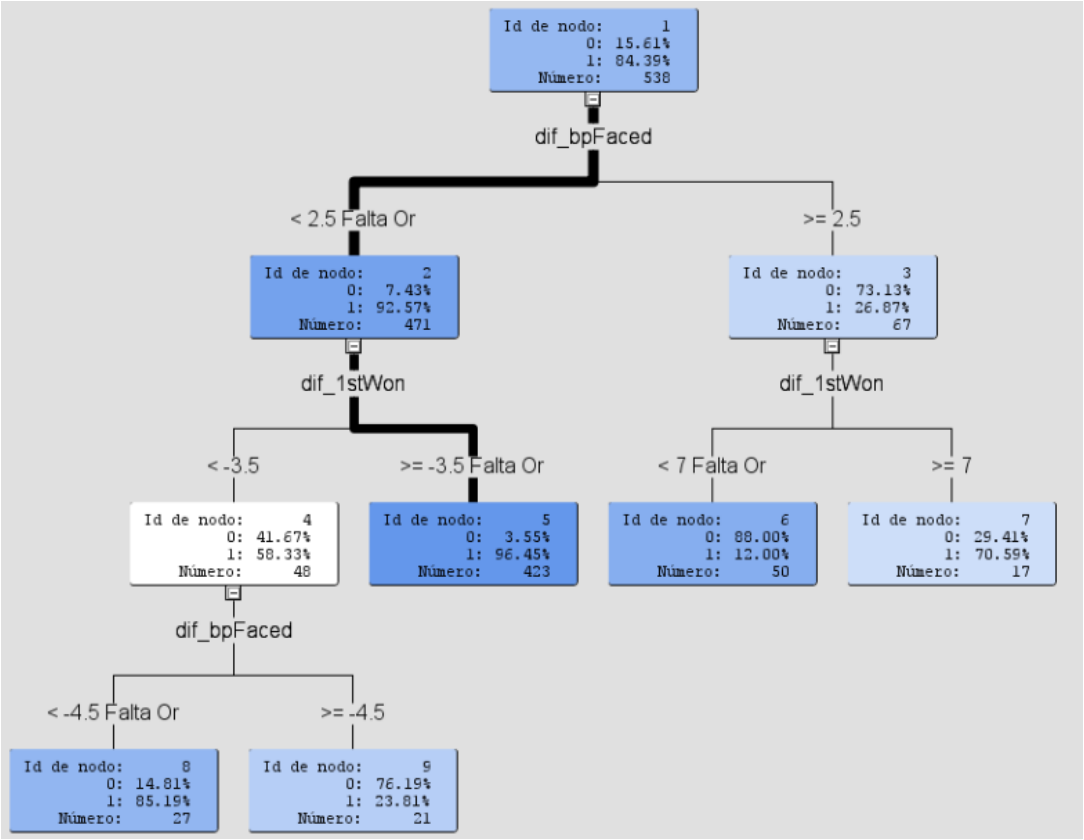


Figura 33. Árbol de clasificación para Rafael Nadal

Analizando un poco los resultados, la diferencia entre los puntos de break de los jugadores (*dif_bpFaced*) juega un papel diferencial para que Rafael Nadal gane el partido. De este modo, si la diferencia en puntos de break es menos de 2.5, el tenista español gana el partido en el 92.57% de las ocasiones. Después, si la diferencia entre los puntos ganados en primer servicio (*dif_1stWon*) es mayor o igual que -3.5, Rafael Nadal gana el partido en el 96.45% de las observaciones estudiadas. Con respecto al nodo número 6, se puede observar que es el caso en el que Rafael Nadal tiene más probabilidades de perder, con un 88% de posibilidad. Esta situación se daría si la diferencia entre los puntos ganados en primer servicio (*dif_1stWon*) es menor que 7, y la diferencia entre los puntos de break en el partido (*dif_bp_Faced*) es mayor que 2.5.

6.6.2. Regresión logística

Se llevará a cabo la regresión logística en R, sobre el conjunto de datos de variables seleccionadas inicialmente. Se comprobarán distintos modelos con selección de variables sobre los que se les aplicará validación cruzada repetida. Estos modelos son los que se muestran en la Tabla 10.

Tabla 10. Modelos para la regresión logística en R

Tipo de modelo	Nombre del modelo
Regresión inicial sin aplicar ningún método de selección de variables.	Modelo 1
Regresión manual seleccionando las variables más importantes.	Modelo 2
Regresión forward con AIC.	Modelo 3
Regresión stepwise con AIC.	Modelo 4
Regresión stepwise con BIC.	Modelo 5
Regresión stepwise con interacciones y AIC.	Modelo 6
Regresión forward con interacciones y AIC.	Modelo 7
Regresión stepwise con interacciones y BIC.	Modelo 8
Regresión forward con interacciones y BIC.	Modelo 9
Regresión LASSO.	LASSO

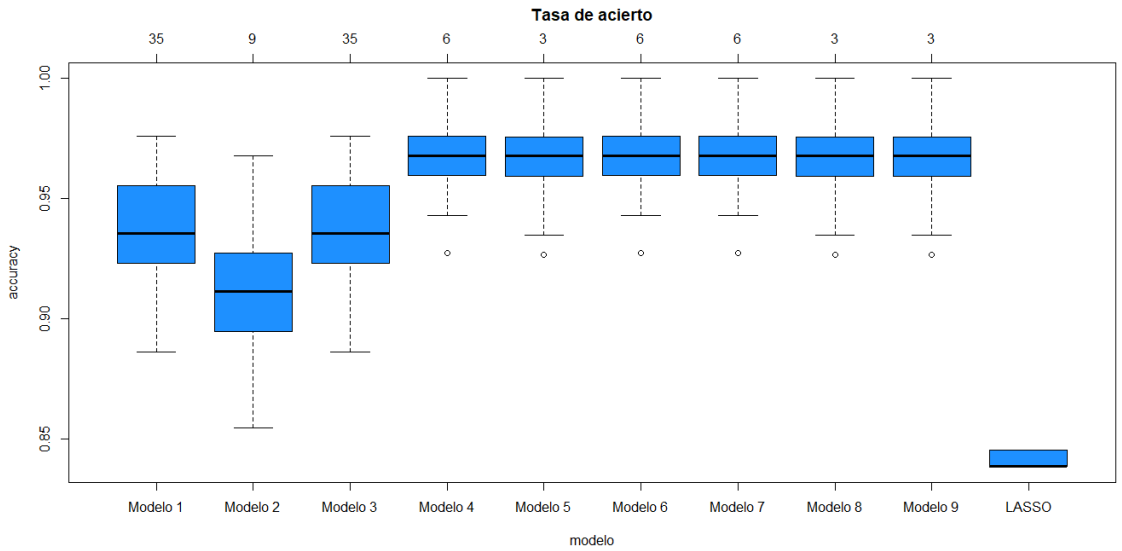


Figura 34. Tasa de acierto de los diferentes modelos de regresión logística.

Para comparar los modelos, se observarán medidas como la tasa de acierto (*accuracy*) y el área bajo la curva ROC. En la Figura 34 se muestran los resultados de la tasa de acierto tras la validación cruzada repetida. Se comprueba que los métodos con stepwise o con interacciones, se suelen ajustar bastante bien, quedando peor clasificadas las 3 primeras regresiones y el LASSO.

En la Figura 35 se muestran los resultados del área bajo la curva ROC.

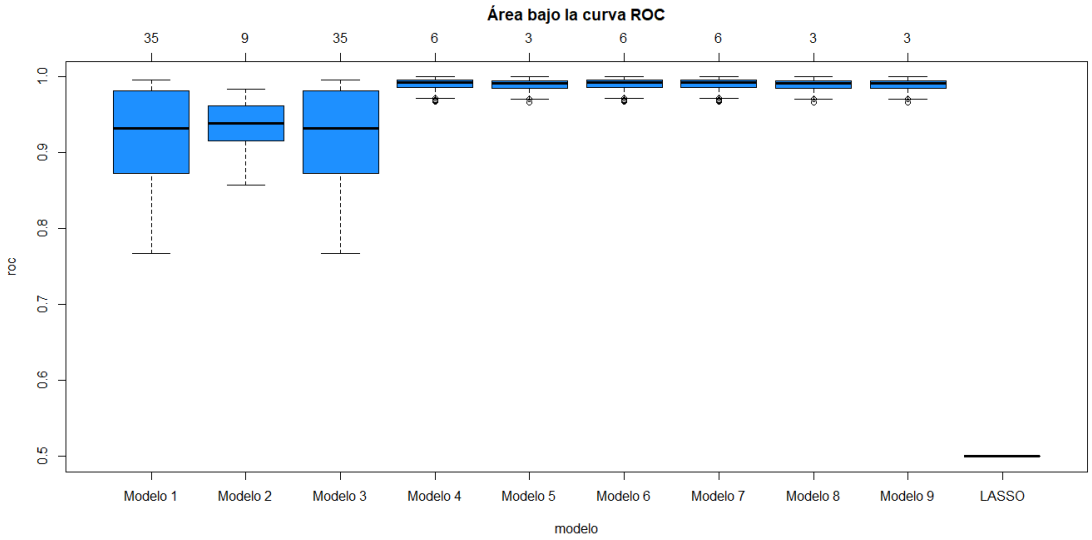


Figura 35. Área bajo la curva ROC de la regresión logística

A priori, tanto en los gráficos de la Figura 34 como el de la Figura 35, se muestra que los modelos 4,5,6,7, 8 y 9 salen muy bien parados en cuanto a las medidas que queremos analizar, siendo todos ellos bastante buenos en cuanto a clasificación se refiere. Sin embargo, dado el principio de parsimonia, se buscará el modelo más sencillo posible. Estos son el 5, el 8 y el 9, que tienen 3 parámetros, frente a los demás, que tienen 6. Si obtenemos la desviación típica y la media en la curva ROC:

Desviación típica:

	modelo	roc
1	Modelo 1	0.062208879
2	Modelo 2	0.030218953
3	Modelo 3	0.062208879
4	Modelo 4	0.008127353
5	Modelo 5	0.008005564
6	Modelo 6	0.008127353
7	Modelo 7	0.008127353
8	Modelo 8	0.008005564
9	Modelo 9	0.008005564
10	LASSO	0.019283764

Media:

	modelo	roc
1	Modelo 1	0.9227389
2	Modelo 2	0.9367892
3	Modelo 3	0.9227389
4	Modelo 4	0.9897353
5	Modelo 5	0.9890755
6	Modelo 6	0.9897353
7	Modelo 7	0.9897353
8	Modelo 8	0.9890755
9	Modelo 9	0.9890755
10	LASSO	0.9791791

El modelo 5 es uno de los que menos desviación presenta. Este modelo selecciona variables a través del método *stepwise* con el criterio BIC, que es más restrictivo a la hora de incluir estas variables en los modelos, por lo que presentando unos altos valores de la curva ROC con muy pocas variables, es el modelo seleccionado. Si estudiamos un poco el modelo ganador, podemos obtener su fórmula:

win_or_lose ~ dif_bpFaced + dif_bpSaved

Además, la importancia de estas variables en el modelo las podemos ver en la Figura 36. Este gráfico quiere decir que si, por ejemplo, se elimina la variable *dif_bpFaced*, pseudo R^2 de McFaden disminuiría en 0.7 unidades, aproximadamente.

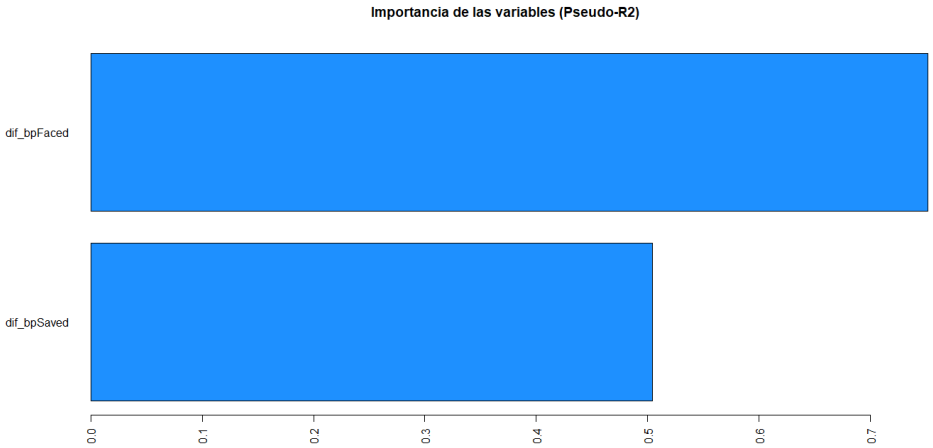


Figura 36. Importancia de las variables en el modelo ganador de regresión logística

Además, se pueden sacar sus parámetros de precisión, sensibilidad, y especificidad:

Accuracy	Sensitivity	Specificity
0.9545455	0.9769231	0.8333333

Y los odds ratio asociados a las variables del modelo:

dif_bpFaced	dif_bpSaved
0.1064034	10.8615976

Es decir, con el modelo seleccionado, se es capaz de explicar con un 95.5% de precisión si Rafael Nadal gana o pierde un partido, atendiendo a un sencillo modelo de regresión con únicamente dos variables que además tiene una muy alta sensibilidad (97.6%). La especificidad (0.83) indica que hay un número bajo de falsos positivos, es decir, una clasificación errónea por parte del modelo a la hora de decir que Rafael Nadal gana un partido cuando en realidad no lo ha hecho. Con respecto a los odds ratio, cabe destacar que el aumento en una unidad de *dif_bpSaved*, y si el resto de las variables se mantuviesen constantes, aumentaría la posibilidad de que Rafael Nadal ganara el partido en 10.86 veces más que si no se aumentara. Con *dif_bpFaced*, al ser menor que 1, se debe hacer su inversa para no malinterpretar el coeficiente. De este modo, el aumento de una unidad en esta variable, la probabilidad de que Rafael Nadal gane el partido aumenta 9.39 veces más (1/0.1064034) que si no se aumentara dicha unidad.

Con respecto al punto de corte, conviene comentar que está fijado en 0.5, y no se prueban otros valores ya que los valores obtenidos son ya lo suficientemente buenos.

6.6.3 Comparación del mejor modelo de clasificación

Con el objetivo de tener más medios de comparación entre los dos métodos de clasificación escogidos, se propone hacer un training-test repetido 20 ocasiones con el árbol ganador y la misma regresión logística estudiada en R en SAS Miner. Para la regresión, se deberá especificar las variables que se han seleccionado en R mediante *Stepwise y BIC* (*dif_bpFaced y dif_bpFaced*) y rechazar todas las demás.

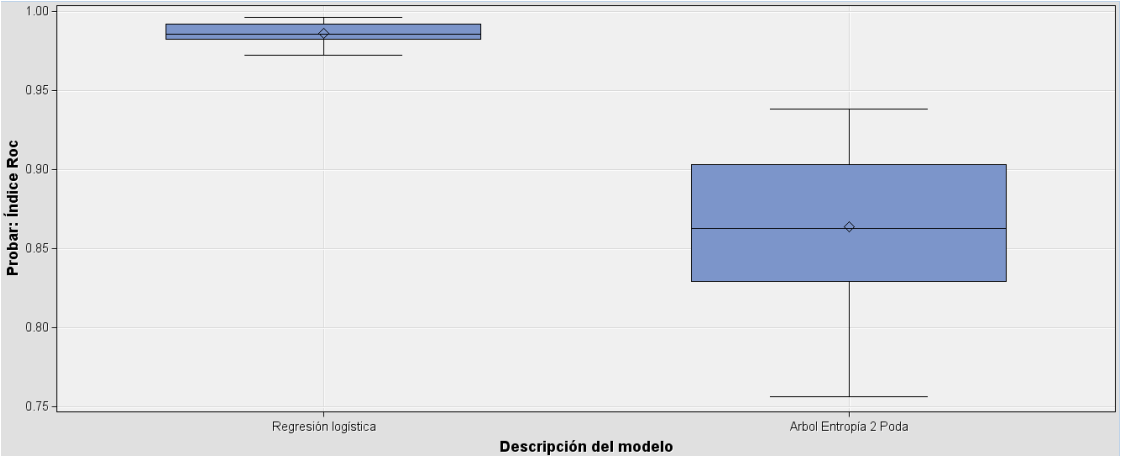


Figura 37. Diagrama de cajas para la comparación entre los modelos explicativos

En la Figura 37 se comprueba que la regresión logística ofrece muchísima más estabilidad en términos de área bajo la curva ROC que el árbol, por lo que este será el modelo elegido, con una media en el índice ROC del **98%** aproximadamente, como ya habíamos visto en R.

7. Análisis y visualización de las predicciones

Como parte final del proyecto se propone hacer análisis sobre los datos de los partidos que se han jugado en el año 2019, al disponer de los datos completos y actualizados en el momento de la realización. Por tanto, se han preparado los archivos de la misma manera que se hizo con los que se obtuvieron los modelos y se obtienen las predicciones. Para obtener las predicciones del primer de los objetivos del trabajo se ha utilizado la función *predict* en R y para el segundo de los objetivos se ha utilizado el nodo de *puntuación* de SAS Miner.

7.1 Visualización de datos: Objetivo I

Para esta visualización, se ha escogido el programa de visualización Tableau. En la Figura 38, se puede observar el cuadro de mando realizado para los resultados obtenidos en la predicción de la duración de un partido de tenis.

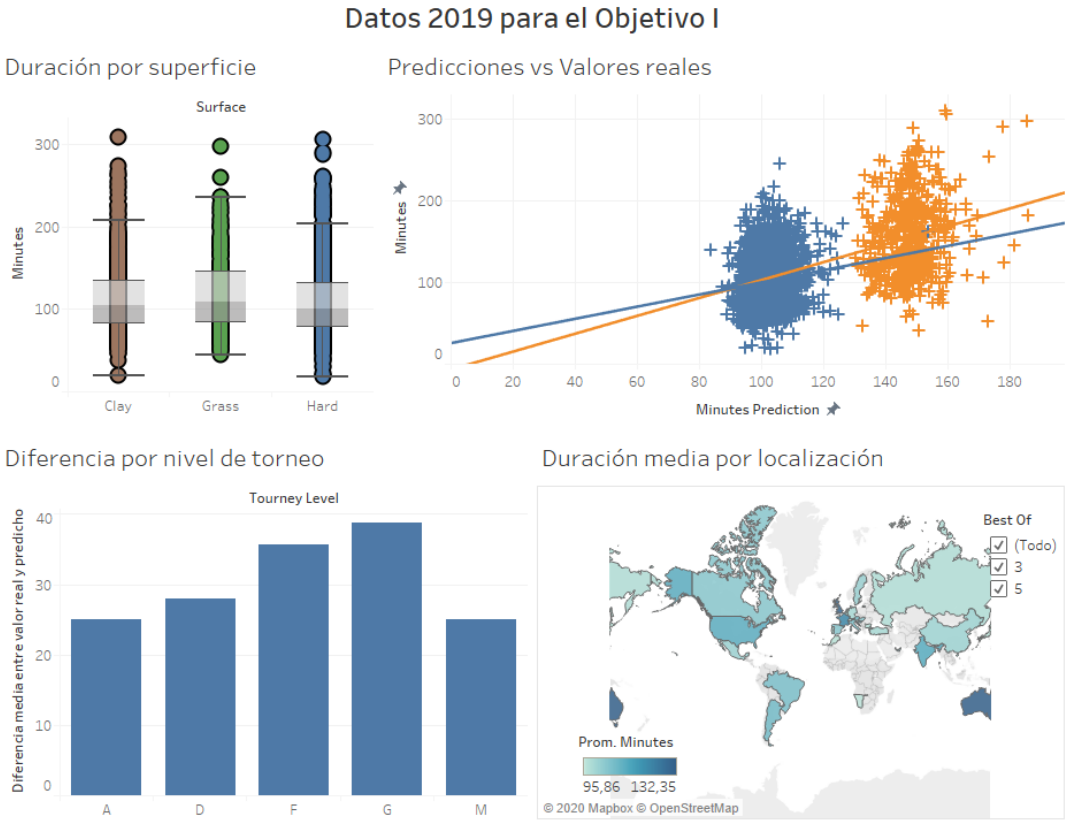


Figura 38. Visualización del cuadro de mando elaborado para el Objetivo I

Ahora bien, se propone separar los gráficos en Figuras individuales para facilitar su visualización.

En el gráfico de la Figura 39 podemos observar la variación de la duración por tipo de superficie para los partidos de 2019. Aunque se vuelve a demostrar que la superficie no influye de manera significativa en la duración de los partidos, se puede comprobar que la duración de los partidos jugados en hierba es algo mayor que en tierra batida y en pista dura. Siendo obvio (y demostrado por los modelos obtenidos) que los partidos que duran más son los de tipo Grand Slam, este resultado puede

deberse a que en Wimbledon (Grand Slam de tipo hierba) existe desde hace tiempo la norma de que los desempates en el último set deben resolverse con una diferencia de 2 juegos, mientras que en los demás Grand Slam, a día de hoy, existe la norma de jugar un *super tie-break*[2] para decidir el ganador en el quinto set.

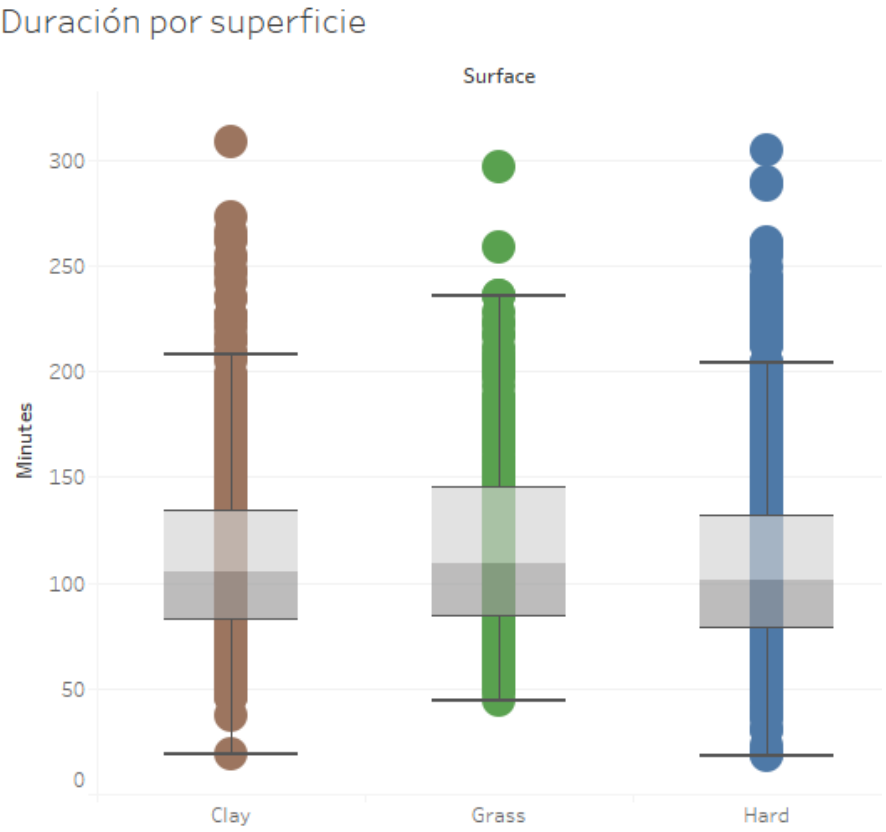


Figura 39. Duración por superficie de los datos del año 2019

En la gráfica de la parte inferior izquierda de la Figura 40 se muestra la diferencia obtenida tras calcular el error entre las predicciones y los valores realmente observados para cada uno de los partidos del año 2019. Lo que se muestra en las columnas corresponde al promedio de los errores. Se observa que la diferencia entre los valores predichos y los reales es mayor en los torneos Grand Slams y en las Finales, ya sean de ATP o Next Gen. Esto refuerza la idea de la dificultad de predecir la duración de un partido de este deporte, más si cabe cuando son partidos en los que hay involucrados jugadores de un nivel muy alto y con muchas situaciones de tensión.

[2] Es un método de desempate en el set decisivo en el que, en caso de empate a 6 juegos en el último set, se juega el desempate a 10 puntos naturales, no siendo necesario que se continúen los partidos hasta que un jugador alcance 2 juegos de diferencia en el último set. Cada vez son más los grandes torneos que se acogen a este método para no alargar de manera excesiva los partidos.

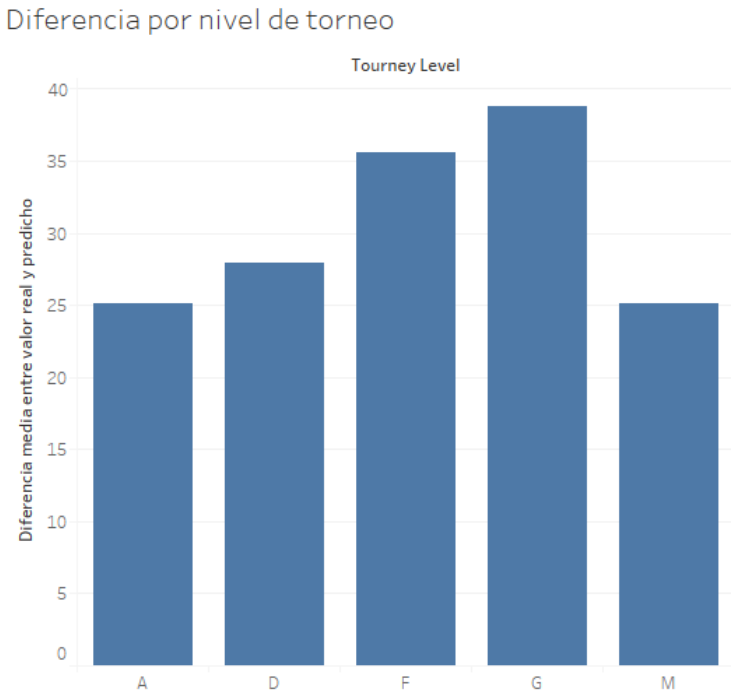


Figura 40. Diferencia entre las predicciones y los valores reales por el nivel del torneo

La calidad de las predicciones se puede comparar en el gráfico de la Figura 41, en el que a través de un gráfico de dispersión se muestran los valores predichos en el eje X y los valores reales en el eje Y. Los resultados se encuentran diferenciados por color dependiendo de si el partido se ha jugado a 5 sets (naranja) o a 3 (azul). Las diagonales nos permiten ver la calidad de la predicción. De este modo, las cruces que están más cerca de la línea se traducen en una buena predicción, mientras que cuanto más lejos esté un valor de la línea diagonal, peor es la predicción que se ha realizado del partido en cuestión.

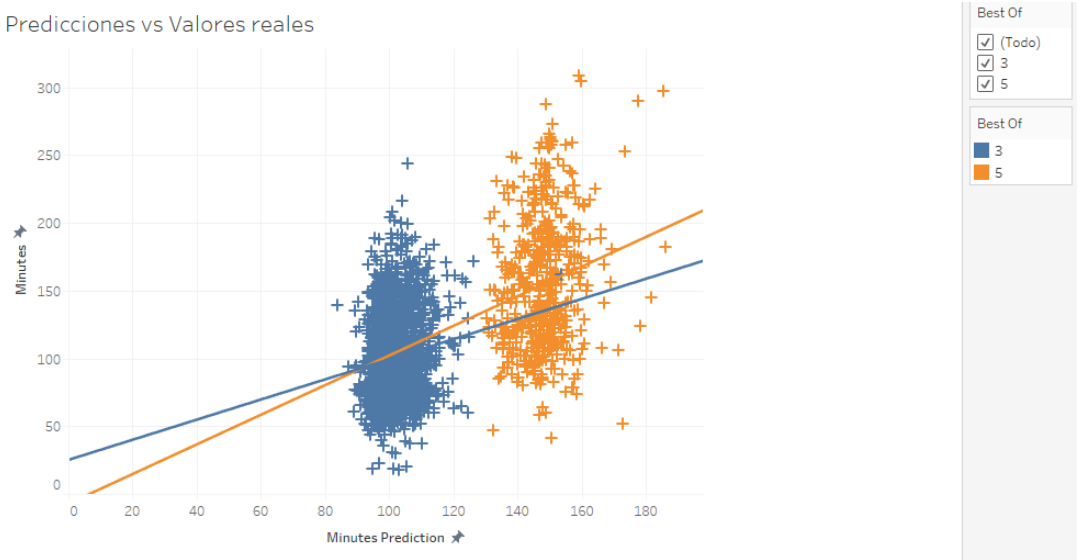


Figura 41. Calidad de las predicciones

Por último, en las Figuras 42 y 43 se muestran dos partes del mismo mapa reflejado en la parte inferior derecha de la Figura 38, con en el que a través de un filtro interactivo se puede comprobar mediante un gradiente la duración media de los torneos jugados en cada país en el año 2019. Como no sería correcto comparar partidos cuyo máximo de sets es al mejor de 3 o de 5, se incluye un filtro con la variable *best_of*, y de esta manera navegar por el mapa comparando partidos de la misma duración, de ahí la separación en las dos Figuras mencionadas anteriormente.

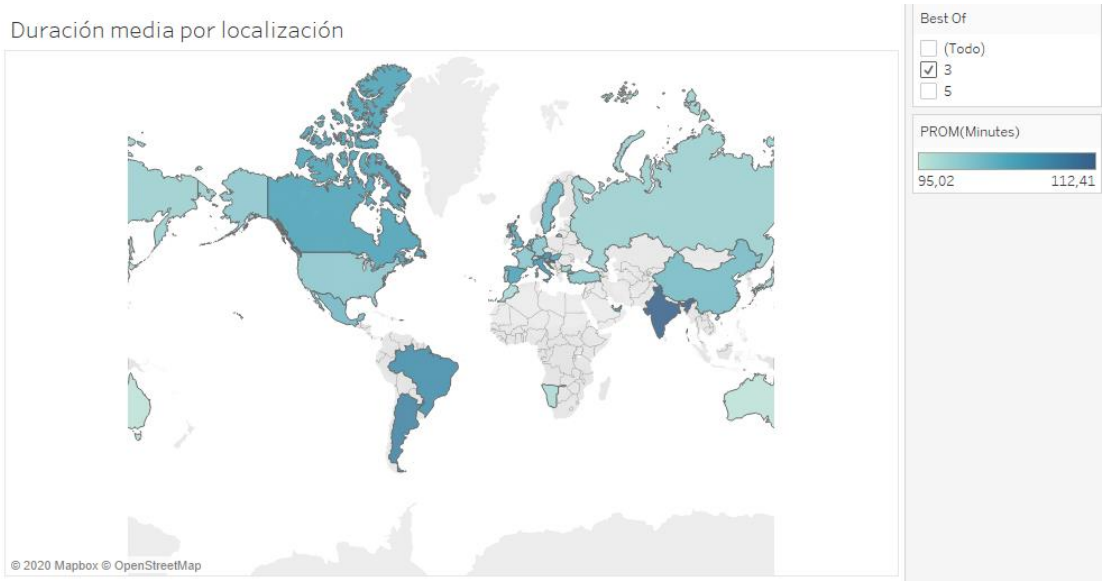


Figura 42. Mapa con la duración media por país. Filtro en best_of: 3

Figura 43. Mapa con la duración media por país. Filtro en best_of: 5

7.2 Visualización de datos: Objetivo II

Para esta parte, la visualización se ha llevado a cabo con Power BI. En la Figura 39 podemos ver el rendimiento del modelo en términos predictivos para la clasificación de los partidos del 2019 de Rafael Nadal en victorias o derrotas. Hay que mencionar que se ha creado un campo calculado en el que, si el valor de la predicción coincide con el real, se contabiliza como acierto, mientras que, si es diferente, aparece como fallo.

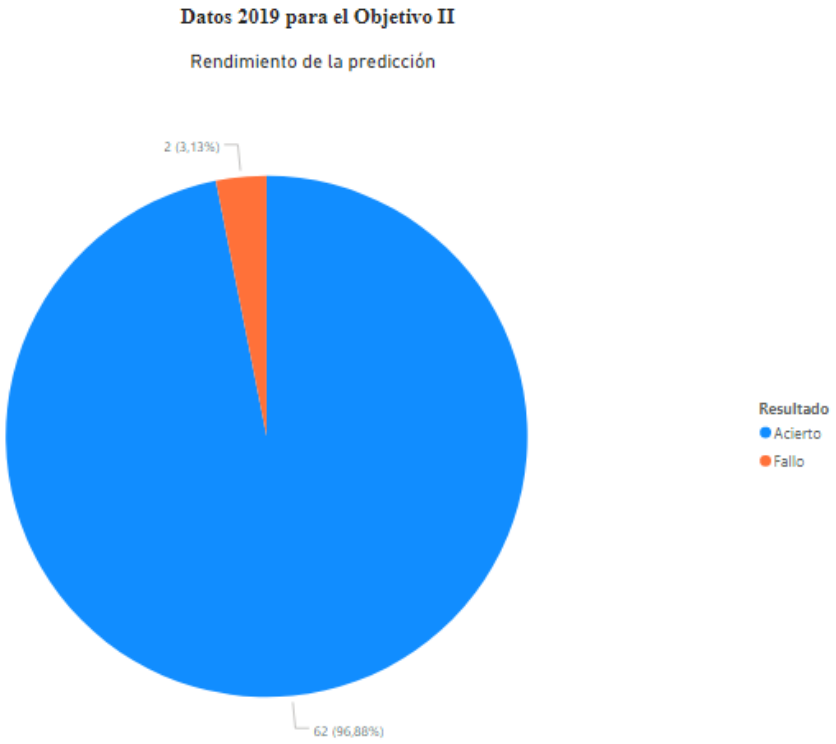


Figura 44. Predicción para el Objetivo II.

Tal y como se puede observar en la Figura 44, la predicción ha sido extremadamente precisa, fallando únicamente en 2 de los 64 partidos que el tenista español ha jugado en el año 2019. En la Tabla 11 se muestra la matriz de confusión, en la que podemos observar qué ocurre en los aciertos o en los fallos del gráfico anterior. Tal y como se puede ver, los fallos del modelo vienen dados por la tasa de falsos positivos, en los que ha identificado una victoria cuando en realidad el evento que se ha producido ha sido una derrota del tenista español

Tabla 11. Matriz de confusión de las predicciones de Rafael Nadal

Matriz de confusión		Predicho	
		Positivos	**Negativos**
Real	**Positivos**	57	0
	Negativos	2	5

Si nos adentramos un poco en la naturaleza de los resultados, se puede observar la tabla que se muestra en la Figura 45.

Resultado	win_or_lose_prediction	tourney_name	surface	tourney_level	month	op_hand	round
Fallo	Ganado	Acapulco	Hard	A	febrero	R	R16
Fallo	Ganado	Wimbledon	Grass	G	julio	R	SF

Figura 45. Observaciones mal clasificadas en el modelo durante la predicción del Objetivo II

A la vista de los resultados en la Figura 45, las observaciones mal clasificadas corresponden a partidos que el modelo ha dado por ganados, cuando en realidad los ha perdido, correspondientes a los torneos de Wimbledon y Acapulco.

8. Conclusiones

Con este estudio se ha pretendido acercar al usuario al análisis de datos y las estadísticas avanzadas al mundo del tenis en particular, a partir de los dos objetivos principales que se han marcado para este trabajo. De igual manera, las conclusiones que se extraen y el trasfondo de la investigación pueden ser aplicables a otros deportes, cuyas entidades organizativas, televisivas o lucrativas puedan tener especial interés en aplicar técnicas de predicción y/o clasificación para conocer aspectos como la duración, o rendimientos individuales de jugadores, como ya se hace en empresas que se dedican a esto, también llamado *Sport Analytics*.

8.1. Conclusiones para el Objetivo I: Predicción de la duración de un partido de tenis

Con respecto a este objetivo, con esta investigación se pretende encontrar un modelo que sea capaz de predecir la duración de un partido de tenis. Como ya se ha comentado en la introducción, este tema no es nuevo si estudiamos los antecedentes en televisiones y casas de apuestas. De los datos de los que se disponen, se extraen dos conclusiones principales con respecto a las variables:

- La variable que más influye en la duración de los partidos es si el partido se juega al mejor de 3 o de 5 sets.

- El tipo de superficie en la que se juega el partido no es relevante para la duración de un partido, al menos no de forma directa. Esto es contrario a la extendida creencia de que hay superficies que *per se* "ralentizan" el tiempo de juego de un partido, como la tierra batida, en la que puede parecer que hay más tiempo de peloteo o el bote de la pelota es más lento.

Los modelos obtenidos no han conseguido predecir con exactitud los minutos que puede durar un partido de tenis. Sin embargo, es conveniente comparar qué ocurre si no se hace ningún modelo. Para ello, se puede conocer cuánto nos equivocamos de media con el modelo seleccionado, y cuál sería el error si no se hiciera nada. Para conseguir esto, se puede usar la fórmula del error absoluto medio, que viene dada por la siguiente expresión:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}$$

donde y_i e $\hat{y}_i$ son los valores reales y predichos, de la variable objetivo respectivamente. De este modo, y como para el no modelo no se tienen los valores de las predicciones, se toma como valor de $\hat{y}_i$ el valor de duración media de la base de datos original. Esta duración es de 111 minutos aproximadamente.

Por tanto, aplicando la fórmula del error absoluto medio obtenemos las medidas de la Tabla 12.

Tabla 12. Error absoluto medio del modelo seleccionado y del no modelo

Medidas aproximadas de los modelos	MAE
No modelo	32,10
Modelo Random Forest seleccionado	27,7

Como se puede observar, la diferencia entre realizar un modelo de predicción o utilizar la duración media como valor de predicción menos de 5 minutos en términos de mejora de error absoluto. Sin embargo, este dato se debe poner en perspectiva. Por ejemplo, en un Grand Slam, el tipo de torneo más prestigioso en el mundo del tenis, y el que más dura en términos de calendario (dos semanas) debido a que todos los partidos se juegan a 5 sets en categoría masculina, se juegan un total de 127 partidos. Por lo tanto, en este contexto, se podría obtener una mejora de 558.8 minutos, aproximadamente. Si se pone en términos de horas, la mejora supondría un total de **9.3 horas**. De esta manera, aunque no se hayan podido obtener buenas medidas de error en los modelos, el modelo podría no ser desdeñable desde el punto de vista organizativo de la propia directiva del torneo y desde la visión de las televisiones para organizar su parrilla de un modo más eficiente. No obstante, como se intuía en base a la literatura consultada, la duración de los partidos de tenis resulta muy complicada de predecir, incluso con modelos complejos de Machine Learning.

8.2. Conclusiones para el Objetivo II: Explicación y clasificación para Rafael Nadal.

En esta segunda línea de investigación, se persigue el objetivo de elaborar un modelo explicativo que permita vislumbrar qué influye en que el tenista español Rafael Nadal gane o no un partido. Antes de realizar cualquier modelo, un pensamiento que se puede llegar a tener es que la variable superficie es la que más puede influir, dado la ya conocida maestría del tenista español en tierra batida. Lo cierto es, que a pesar de ser el tipo de torneo en el que más torneos ha conseguido, Rafael Nadal es un tenista que de por sí gana bastantes partidos en cualquiera de las superficies en las que se juegue un torneo. Fruto de esto es que la variable *surface* queda eliminada en la selección de variables sin apenas tener importancia para la variable objetivo. Por tanto, se realizan modelos de clasificación interpretables (regresión logística y árboles de clasificación) que permitan luego explicar con facilidad en qué momentos gana Rafael Nadal el partido o cuando lo ha perdido y por qué. Con respecto a esto, se extraen varias conclusiones.

- El mejor modelo de clasificación se obtiene mediante una regresión logística penalizando al máximo la entrada de nuevas variables en el modelo mediante la selección *Stepwise* y el criterio de selección bayesiano o *BIC.* En este modelo se consigue tener un 95% de tasa de acierto atendiendo únicamente a dos variables.

- Con respecto a las variables, que son la diferencia de los puntos de break a los que se enfrentan los jugadores, y la diferencia entre los puntos de break que salvan cada uno de ellos, es digno de mención que se trata de variables que no existían en el conjunto de datos original, sino que se han creado a partir de las estadísticas individuales de los jugadores en cada uno de los partidos. Estas variables influyen en gran medida en las posibilidades de que ocurra el evento, es decir, de que Rafael Nadal gane el partido, tal y como se ha visto en el estudio de los *odss ratio*, con un valor de 10.86 para *dif_bpSaved* y de 9.39 para *dif_bpFaced.*

- Como se ha podido observar en el rendimiento de la predicción elaborado con Power BI, el modelo es bastante preciso, habiendo clasificado correctamente

todos los partidos ganados por Rafael Nadal, y fallando únicamente en 2 de los 7 partidos que aparecían en la base de datos como perdidos. Esto tiene una gran importancia, debido a que, a priori, puede resultar sencillo clasificar los partidos del tenista español, ya que suele tener un porcentaje de victorias elevado. Sin embargo, con los resultados obtenidos en la predicción, tenemos un 71,4 % de acierto en el caso del no evento, es decir, en la derrota de Rafael Nadal, por lo que se ha conseguido un modelo que además de ser bastante preciso, es en gran medida, lo suficientemente específico como para clasificar las derrotas de Rafael Nadal de un modo fiable.

9.Bibliografía

Abbas, N. M. (2019, agosto 21). *NBA Data Analytics: Changing the Game*. Medium. https://towardsdatascience.com/nba-data-analytics-changing-the-game-a9ad59d1f116

Calviño Martínez, A. (2019). *Apuntes en Técnicas de Minería de Datos*. Facultad de Estudios Estadísticos. Universidad Complutense de Madrid.

El Español. (2018, diciembre 18). *Copa Davis: Impacto económico de 40 millones de euros y 600 puestos de trabajo*. El Español. https://www.elespanol.com/deportes/tenis/20181218/copa-davis-impacto-economico-millones-puestos-trabajo/361714597_0.html

Gorgi, P., Koopman, S. J., & Lit, R. (2018). The Analysis and Forecasting of ATP Tennis Matches Using a High-Dimensional Dynamic Model. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3110555

Kovalchik, S. & Ingram, M. (2018). Estimating the duration of professional tennis matches for varying formats. Journal of Quantitative Analysis in Sports, 14(1), pp. 13-23. Retrieved 11 Dec. 2019, from doi:10.1515/jqas-2017-0077

KU Leuven. (2020). *Machine Learning and Data Mining for Sports Analytics*. https://dtai.cs.kuleuven.be/events/MLSA20/links.php

LUCA. (s. f.). *Sports Analytics y Cultura: Uniendo deporte, ocio y tecnología*. Sports Analytics y Cultura: uniendo deporte, ocio y tecnología. Recuperado 6 de diciembre de 2019, de https://luca-d3.com/es/industrias/ocio-tecnologia

Luque, D. G. T. (2014). *Características de la estructura temporal en tenis. Una revisión.Universidad de Jaén Facultad de Humanidades y Ciencias de la Educación.* 12.

Martin, C., Thevenet, D., Zouhal, H., Mornet, Y., Delès, R., Crestel, T., Ben Abderrahman, A., & Prioux, J. (2011). Effects of Playing Surface (Hard and Clay Courts) on Heart Rate and Blood Lactate During Tennis Matches Played by High-Level Players: *Journal of Strength and Conditioning Research*, *25*(1), 163-170. https://doi.org/10.1519/JSC.0b013e3181fb459b

Nieto Agudo, T. (2015). *Aplicación de técnicas de minería de datos a la predicción de partidos de tenis*.

Portela García-Miguel, J. (s. f.). *Apuntes de la asignatura «Machine Learning»*. Facultad de Estudios Estadísticos. Universidad Complutense de Madrid.

Puntodebreak. (s. f.). *La duración de los partidos de tenis*. Puntodebreak. Recuperado 8 de febrero de 2020, de http://www.puntodebreak.com/2012/02/15/la-duracion-de-los-partidos-de-tenis

SAP. (2019, octubre 16). *New Feature from SAP and WTA: Patterns of Play*. SAP News Center. https://news.sap.com/2019/10/sap-wta-launch-new-patterns-of-play-feature/

Universidad Europea. (2019). *La Universidad Europea lidera el II Estudio sobre el impacto social y económico del torneo de tenis Mutua Madrid Open*. Universidad Europea. https://universidadeuropea.es/noticias/la-universidad-europea-lidera-el-ii-estudio-sobre-el-impacto-social-y-economico-del-torneo-de-tenis-mutua-madrid-open

Web Apuestas. (s. f.). *Apuestas de la duración de un partido de tenis | Web Apuestas*. Recuperado 8 de febrero de 2020, de https://www.webapuestas.com/taxonomy/term/84/undefined

ANEXOS

A. Conjunto de datos de predicción de la duración de un partido de tenis

A.1 Tablas y Figuras

Tabla 1: Análisis de la variable de clase.

Nombre de la variable	Nivel	CODE	Número de ocurrencias	Tipo	Porcentaje
best_of	3	0	11987	N	79.27386
best_of	5	1	3134	N	20.72614
month	febrero	1	2278	C	15.06514
month	mayo	4	1810	C	11.97011
month	julio	6	1749	C	11.5667
month	abril	3	1565	C	10.34984
month	agosto	7	1473	C	9.741419
month	enero	0	1419	C	9.3843
month	octubre	9	1384	C	9.152834
month	marzo	2	1242	C	8.213742
month	junio	5	1152	C	7.618544
month	septiembre	8	789	C	5.217909
month	diciembre	11	169	C	1.117651
month	noviembre	10	91	C	0.601812
players_hand	RR	0	10754	C	71.11963
players_hand	RL	1	1866	C	12.34045
players_hand	LR	3	1743	C	11.52702
players_hand	LL	2	297	C	1.964156
players_hand	RU	4	257	C	1.699623
players_hand	UR	6	87	C	0.575359
players_hand	LU	7	35	C	0.231466
players_hand	UU	5	27	C	0.17856
players_hand	R	9	24	C	0.15872
players_hand	UL	8	21	C	0.13888
players_hand	U	10	8	C	0.052907
players_hand		12	1	C	0.006613
players_hand	L	11	1	C	0.006613
round	R32	0	4926	C	32.57721
round	R16	1	2829	C	18.70908
round	R64	6	2321	C	15.34951
round	R128	5	1792	C	11.85107
round	QF	2	1410	C	9.32478
round	RR	7	763	C	5.045963
round	SF	3	716	C	4.735137
round	F	4	362	C	2.394022
round	BR	8	2	C	0.013227
surface	Hard	0	8566	C	56.64969
surface	Clay	1	4902	C	32.41849
surface	Grass	2	1569	C	10.3763
surface	None	3	73	C	0.482772
surface	Carpet	4	11	C	0.072747
tourney_level	A	0	8445	C	55.84948
tourney_level	M	3	3131	C	20.7063
tourney_level	G	1	2784	C	18.41148
tourney_level	D	2	701	C	4.635937
tourney_level	F	4	60	C	0.396799

Tabla 2: Variables de grupo tras selección de variables

NAME	GROUP	VARIABLE	LEVEL
G_tourney_name	0	tourney_name	ACAPULCO
G_tourney_name	1	tourney_name	ANTALYA
G_tourney_name	1	tourney_name	ANTWERP
G_tourney_name	0	tourney_name	ATLANTA
G_tourney_name	0	tourney_name	AUCKLAND
G_tourney_name	4	tourney_name	AUSTRALIAN OPEN
G_tourney_name	0	tourney_name	BANGKOK
G_tourney_name	1	tourney_name	BARCELONA
G_tourney_name	0	tourney_name	BASEL
G_tourney_name	0	tourney_name	BASTAD
G_tourney_name	0	tourney_name	BEIJING
G_tourney_name	0	tourney_name	BOGOTA
G_tourney_name	1	tourney_name	BRISBANE
G_tourney_name	1	tourney_name	BUCHAREST
G_tourney_name	1	tourney_name	BUDAPEST
G_tourney_name	1	tourney_name	BUENOS AIRES
G_tourney_name	0	tourney_name	CABO SAN LUCAS
G_tourney_name	1	tourney_name	CANADA MASTERS
G_tourney_name	1	tourney_name	CASABLANCA
G_tourney_name	1	tourney_name	CHENGDU
G_tourney_name	0	tourney_name	CHENNAI
G_tourney_name	1	tourney_name	CINCINNATI MASTERS
G_tourney_name	2	tourney_name	DAVIS CUP

G_tourney_name	0	tourney_name	DELRAY BEACH
G_tourney_name	0	tourney_name	DOHA
G_tourney_name	0	tourney_name	DUBAI
G_tourney_name	0	tourney_name	DUSSELDORF
G_tourney_name	0	tourney_name	EASTBOURNE
G_tourney_name	1	tourney_name	ESTORIL
G_tourney_name	0	tourney_name	GENEVA
G_tourney_name	1	tourney_name	GSTAAD
G_tourney_name	0	tourney_name	HALLE
G_tourney_name	1	tourney_name	HAMBURG
G_tourney_name	1	tourney_name	HOUSTON
G_tourney_name	1	tourney_name	INDIAN WELLS MASTERS
G_tourney_name	1	tourney_name	ISTANBUL
G_tourney_name	1	tourney_name	KITZBUHEL
G_tourney_name	0	tourney_name	KUALA LUMPUR
G_tourney_name	1	tourney_name	LONDON
G_tourney_name	0	tourney_name	LOS CABOS
G_tourney_name	1	tourney_name	LYON
G_tourney_name	1	tourney_name	MADRID MASTERS
G_tourney_name	1	tourney_name	MARRAKECH
G_tourney_name	0	tourney_name	MARSEILLE
G_tourney_name	0	tourney_name	MEMPHIS
G_tourney_name	0	tourney_name	METZ
G_tourney_name	1	tourney_name	MIAMI MASTERS

G_tourney_name	1	tourney_name	MONTE CARLO MASTERS
G_tourney_name	0	tourney_name	MONTPELLIER
G_tourney_name	0	tourney_name	MOSCOW
G_tourney_name	1	tourney_name	MUNICH
G_tourney_name	1	tourney_name	NEW YORK
G_tourney_name	0	tourney_name	NEWPORT
G_tourney_name	0	tourney_name	NEXT GEN FINALS
G_tourney_name	1	tourney_name	NICE
G_tourney_name	0	tourney_name	NOTTINGHAM
G_tourney_name	1	tourney_name	PARIS MASTERS
G_tourney_name	1	tourney_name	PUNE
G_tourney_name	0	tourney_name	QUEEN'S CLUB
G_tourney_name	1	tourney_name	QUITO
G_tourney_name	1	tourney_name	RIO OLYMPICS
G_tourney_name	1	tourney_name	RIO DE JANEIRO
G_tourney_name	4	tourney_name	ROLAND GARROS
G_tourney_name	1	tourney_name	ROME MASTERS
G_tourney_name	0	tourney_name	ROTTERDAM
G_tourney_name	0	tourney_name	SAN JOSE
G_tourney_name	1	tourney_name	SANTIAGO
G_tourney_name	1	tourney_name	SAO PAULO
G_tourney_name	0	tourney_name	SHANGHAI MASTERS
G_tourney_name	1	tourney_name	SHENZHEN
G_tourney_name	0	tourney_name	SOFIA

G_tourney_name	1	tourney_name	ST. PETERSBURG
G_tourney_name	0	tourney_name	ST.PETERSBURG
G_tourney_name	1	tourney_name	STOCKHOLM
G_tourney_name	0	tourney_name	STUTTGART
G_tourney_name	1	tourney_name	SYDNEY
G_tourney_name	1	tourney_name	TOKYO
G_tourney_name	1	tourney_name	TOUR FINALS
G_tourney_name	4	tourney_name	US OPEN
G_tourney_name	1	tourney_name	UMAG
G_tourney_name	0	tourney_name	VALENCIA
G_tourney_name	1	tourney_name	VIENNA
G_tourney_name	1	tourney_name	VINA DEL MAR
G_tourney_name	0	tourney_name	WASHINGTON
G_tourney_name	3	tourney_name	WIMBLEDON
G_tourney_name	0	tourney_name	WINSTON SALEM
G_tourney_name	0	tourney_name	WINSTON-SALEM
G_tourney_name	1	tourney_name	ZAGREB
G_tourney_name	0	tourney_name	S-HERTOGENBOSCH

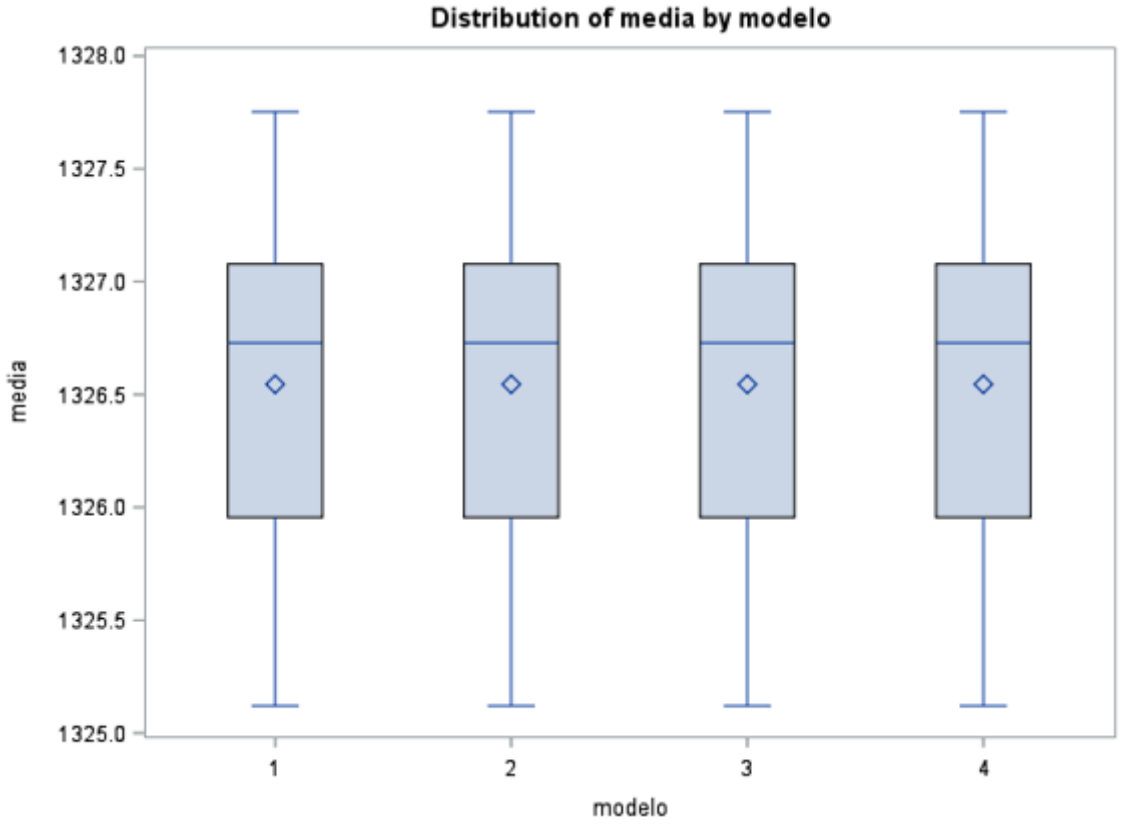
Tabla 3: Modelos más repetidos en la macro ***%randomselect:***

Variables	Repeticiones	Método
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_dif_rank_poi IMP_REP_min_age IMP_REP_min_rank aleat LOG_IMP_REP_max_rank PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	AIC
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_min_age IMP_REP_min_rank aleat PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	AIC
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_min_rank aleat PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	AIC
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_dif_rank_poi IMP_REP_min_age IMP_REP_min_rank aleat LOG_IMP_REP_max_rank PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	BIC
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_min_age IMP_REP_min_rank aleat PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	BIC
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_min_rank aleat PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	BIC
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_dif_rank_poi IMP_REP_min_age IMP_REP_min_rank aleat LOG_IMP_REP_max_rank PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	glmselect
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_min_age IMP_REP_min_rank aleat PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	glmselect
tourney_level best_of round min_rank_points G_tourney_name IMP_REP_min_rank aleat PWR_IMP_REP_min_age LOG_IMP_REP_min_rank	5	glmselect

A.2 Principales salidas de SAS

En este apartado se incluyen algunos de los gráficos obtenidos en la modelización de las redes neuronales con SAS Base, al haber incluido mayoritariamente R en el cuerpo del trabajo, debido a las posibilidades que nos ofrecía en el momento de la realización de este.

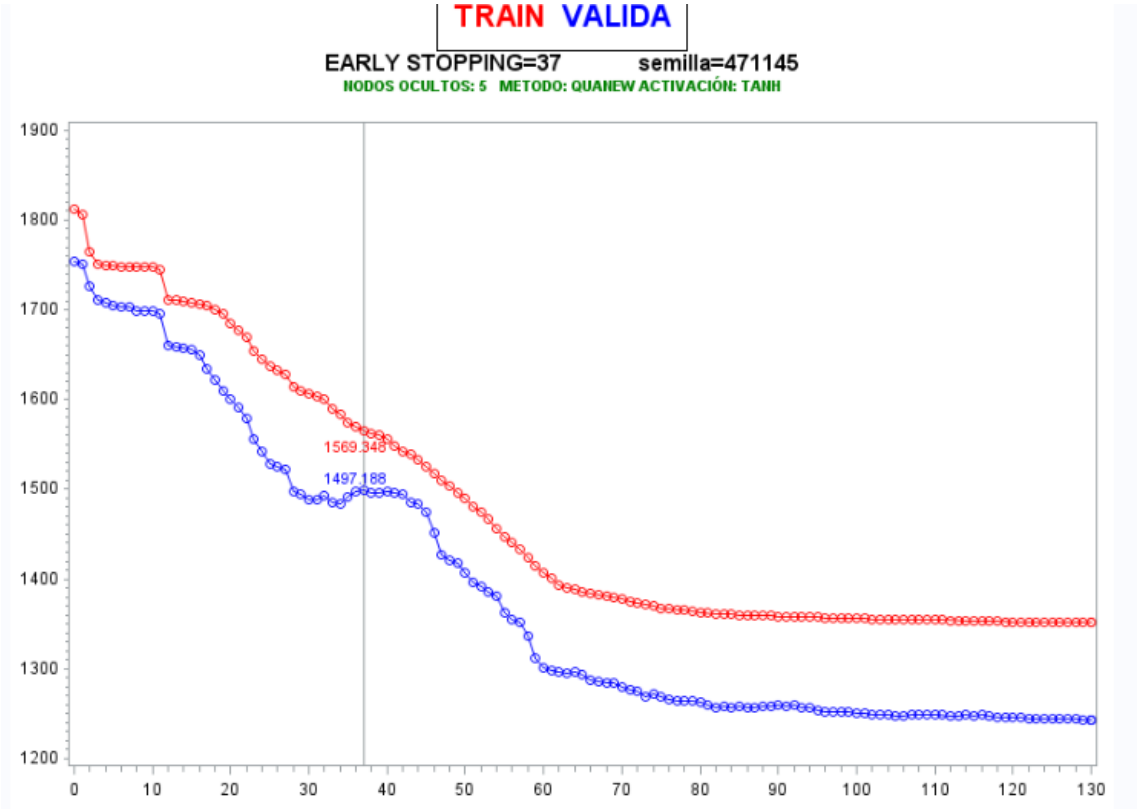
Validación cruzada repetida para los modelos de regresión lineal (macro **%cruzada**):



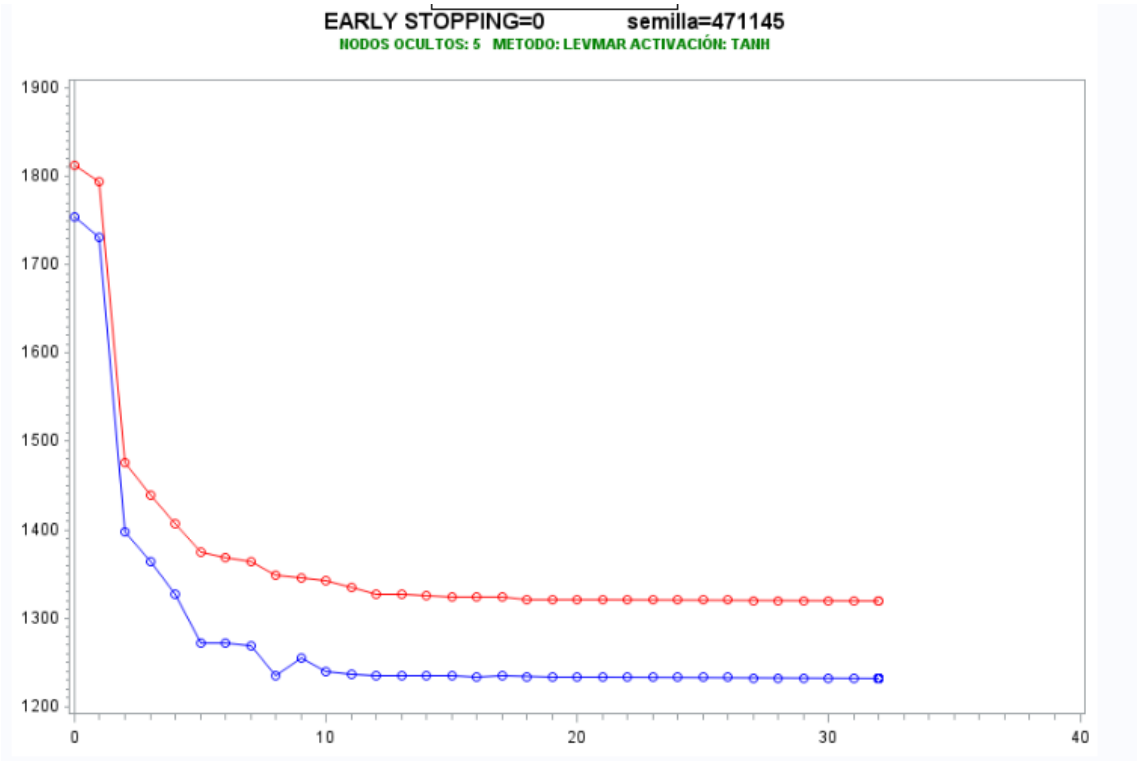
Para estudiar el Early Stopping de una red neuronal, se puede usar la macro **%redneuronal**, e introducir las variables deseadas, así como el número de nodos, el algoritmo de optimización y la función de activación.

Early Stopping con Back Propagation, momentum de 0.5 y learning rate de 0.1 y función de activación tangente hiperbólica:

Early Stopping con QUANEW y función de activación tangente hiperbólica:

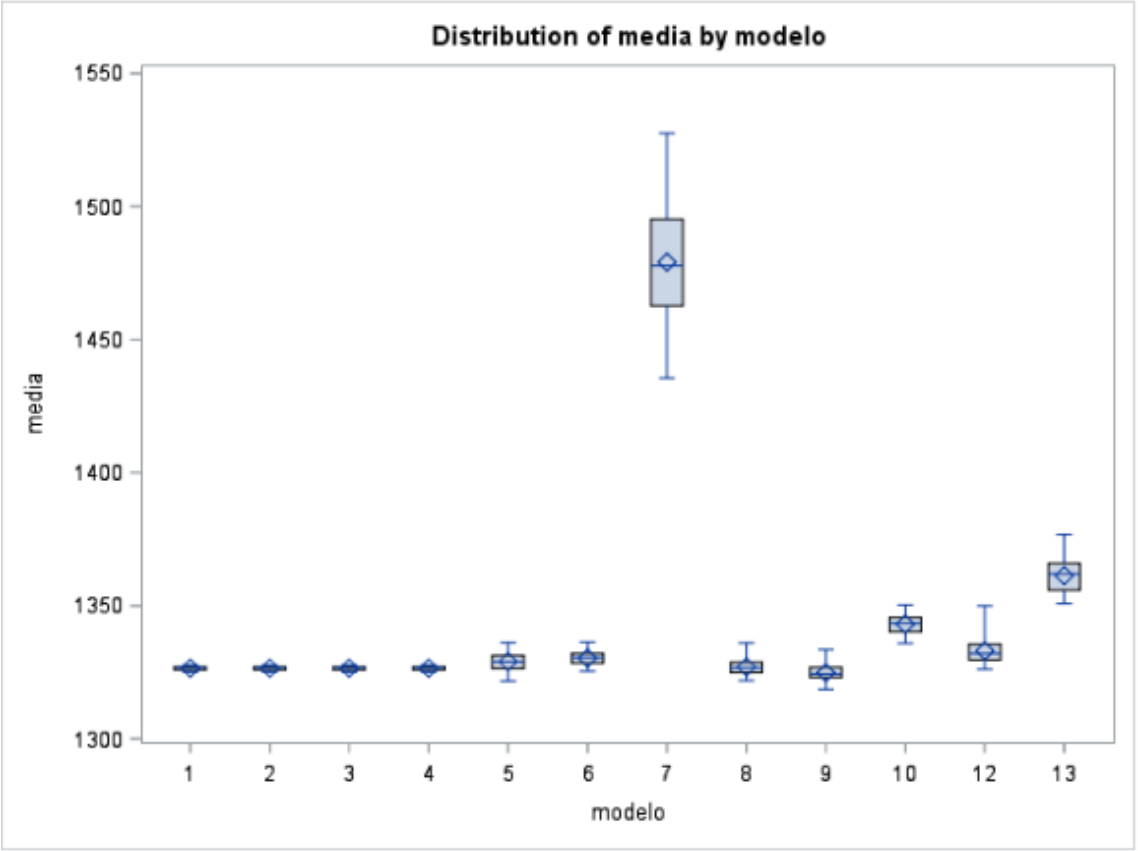


Early Stopping con LEVMAR y función de activación tangente hiperbólica:

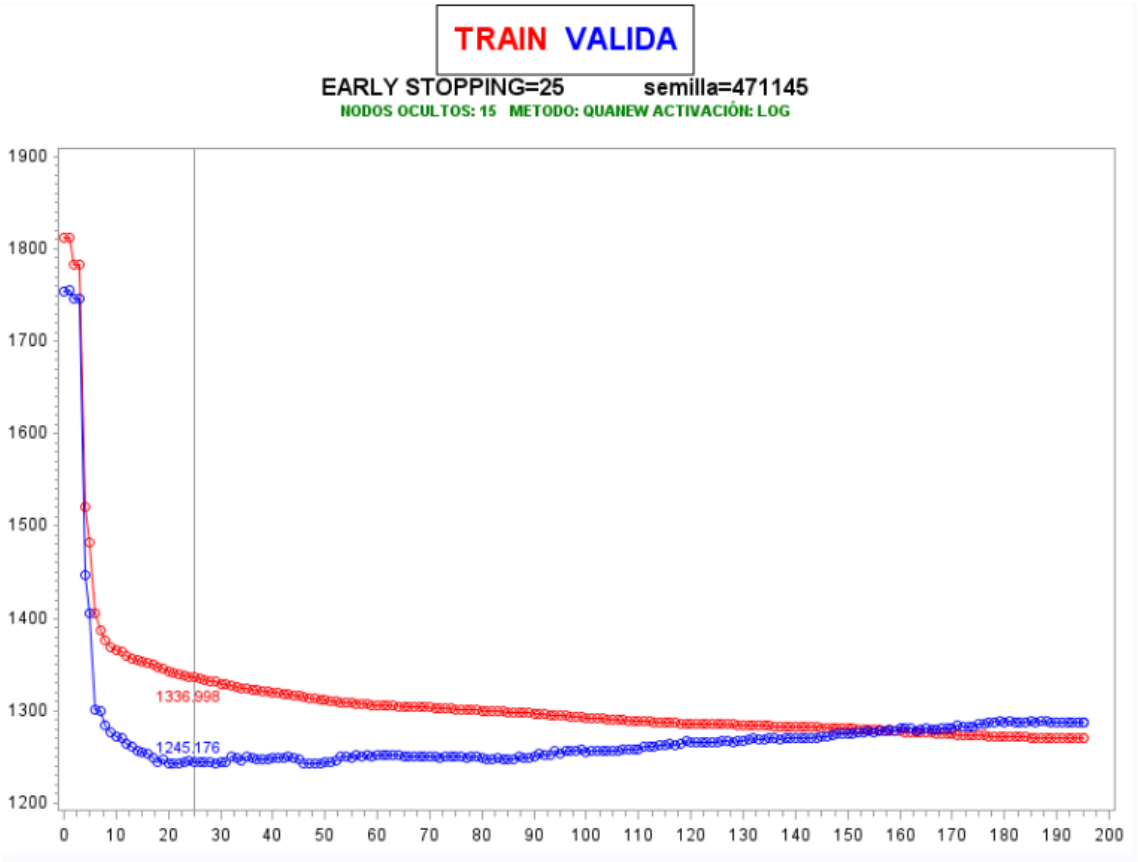


Para probar los modelos en validación cruzada repetida, se usa la macro **%cruzadaneural**.

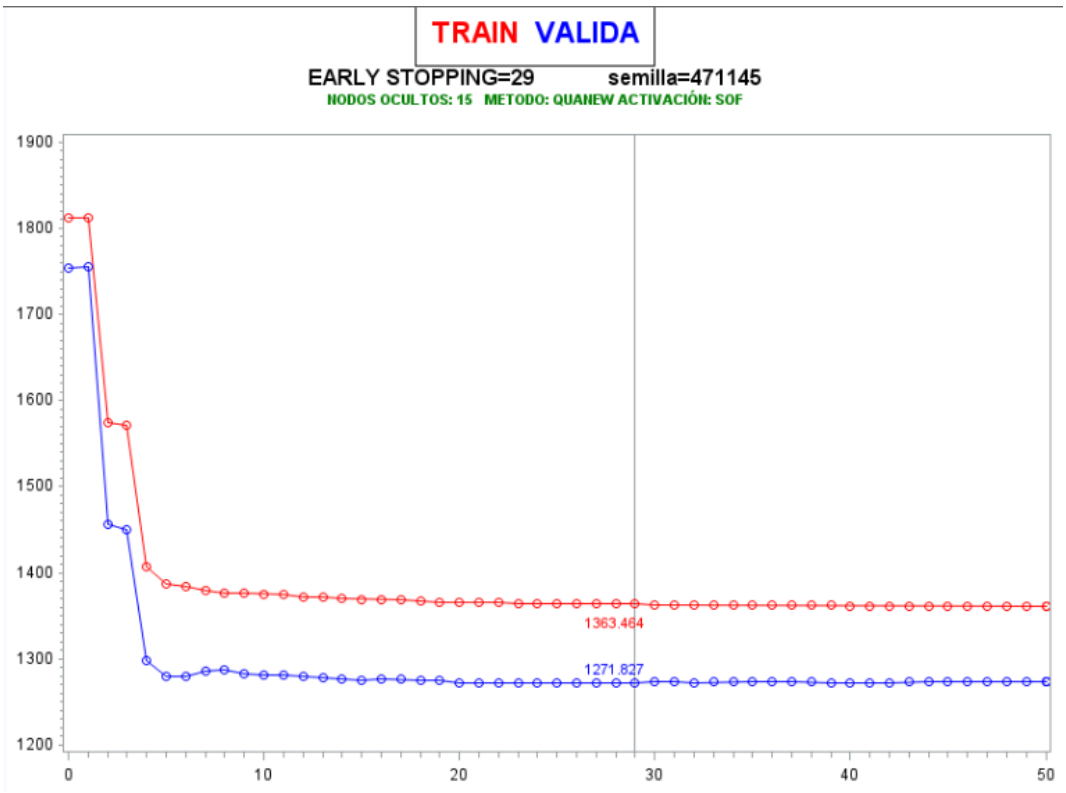
Primer intento para mejorar la regresión lineal en validación cruzada repetida:



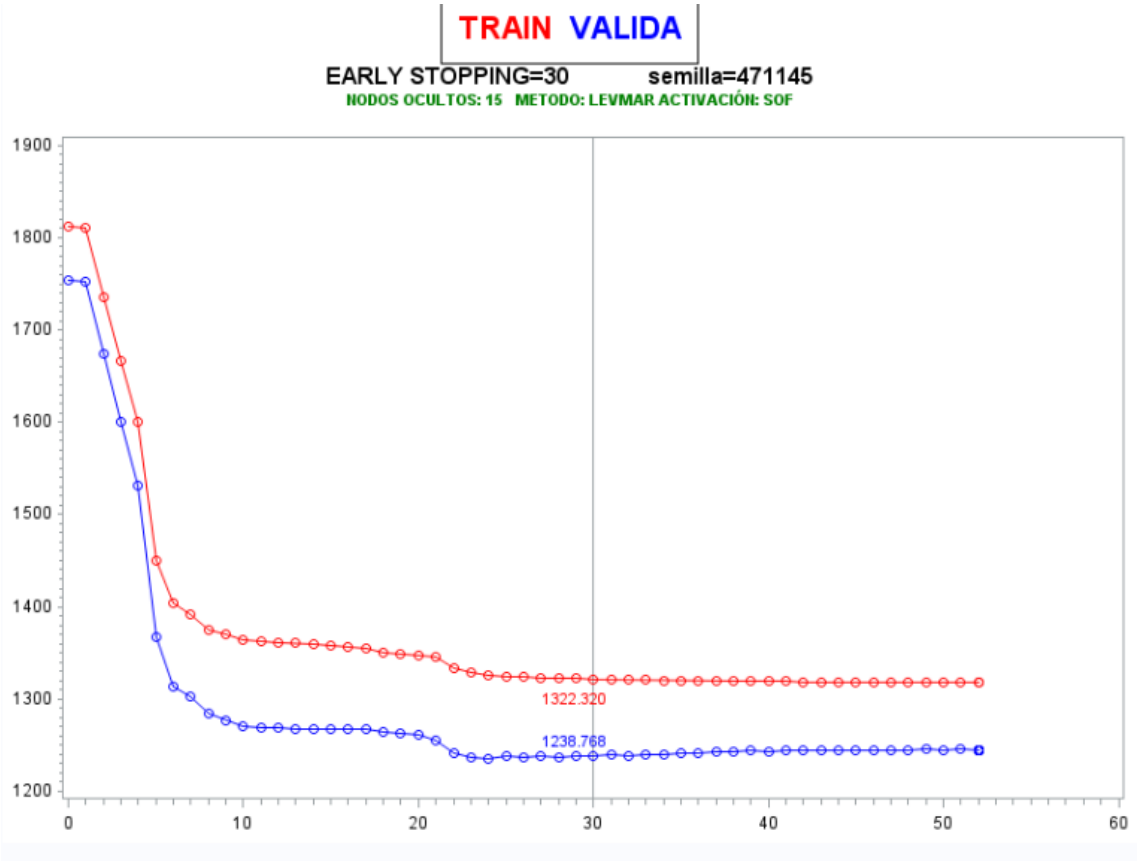
Early Stopping con QUANEW y función de activación logarítmica:



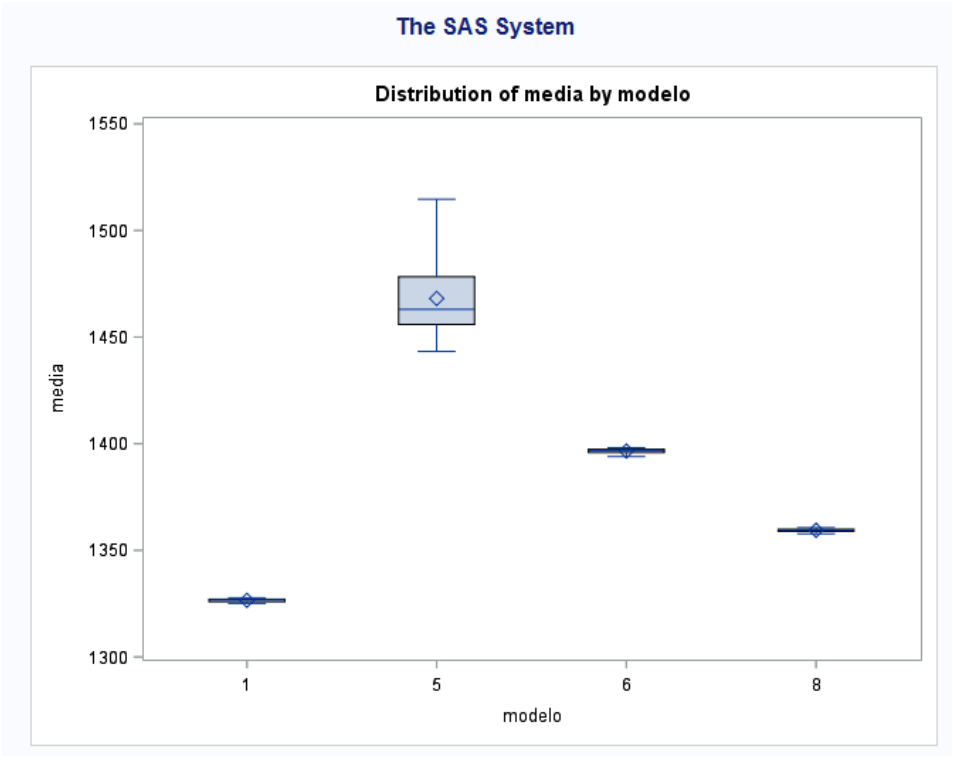
Early Stopping con QUANEW y función de activación Softmax (SOF):



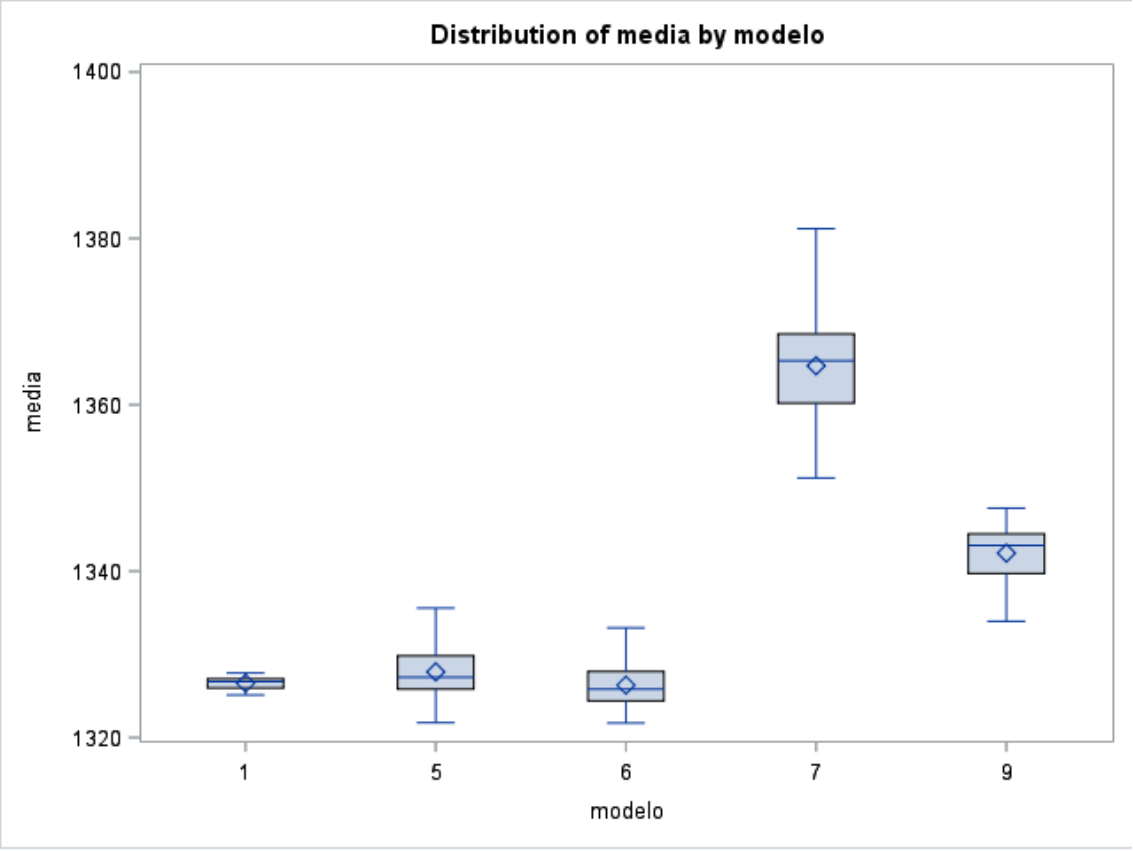
Early Stopping con LEVMAR y función de activación SOF:



Segundo intento de mejora de la regresión lineal en validación cruzada repetida:



Tercer intento de mejora de la regresión lineal en validación cruzada repertida:



En este último caso, se puede ver como los modelos 5 y 6 de redes neuronales, consiguen igualar al de regresión lineal (1) en términos de error, medido en ASE. Sin embargo, debido a que son menos estables y más complejos de por sí, no se puede escoger con seguridad los modelos de redes neuronales obtenidos frente a la regresión lineal.

B. Conjunto de datos de Rafael Nadal.

B.1 Tablas y Figuras

Tabla 4. Variables de grupo para el conjunto de datos de Rafael Nadal

NAME	GROUP	VARIABLE	LEVEL
G_round	4	round	BR
G_round	3	round	F
G_round	1	round	QF
G_round	0	round	R128
G_round	0	round	R16
G_round	0	round	R32
G_round	0	round	R64
G_round	2	round	RR
G_round	2	round	SF
G_tourney_name	1	tourney_name	ACAPULCO
G_tourney_name	2	tourney_name	AUSTRALIAN OPEN
G_tourney_name	5	tourney_name	BANGKOK
G_tourney_name	0	tourney_name	BARCELONA
G_tourney_name	4	tourney_name	BASEL
G_tourney_name	3	tourney_name	BEIJING
G_tourney_name	0	tourney_name	BEIJING OLYMPICS
G_tourney_name	5	tourney_name	BRISBANE
G_tourney_name	2	tourney_name	BUENOS AIRES
G_tourney_name	2	tourney_name	CANADA MASTERS
G_tourney_name	3	tourney_name	CHENNAI
G_tourney_name	4	tourney_name	CINCINNATI MASTERS
G_tourney_name	0	tourney_name	DAVIS CUP WG QF: ESP VS GER
G_tourney_name	3	tourney_name	DOHA
G_tourney_name	5	tourney_name	DUBAI
G_tourney_name	7	tourney_name	HALLE

G_tourney_name	0	tourney_name	HAMBURG
G_tourney_name	0	tourney_name	HAMBURG MASTERS
G_tourney_name	2	tourney_name	INDIAN WELLS MASTERS
G_tourney_name	2	tourney_name	MADRID MASTERS
G_tourney_name	3	tourney_name	MIAMI MASTERS
G_tourney_name	0	tourney_name	MONTE CARLO MASTERS
G_tourney_name	4	tourney_name	PARIS MASTERS
G_tourney_name	4	tourney_name	QUEEN'S CLUB
G_tourney_name	5	tourney_name	RIO OLYMPICS
G_tourney_name	2	tourney_name	RIO DE JANEIRO
G_tourney_name	0	tourney_name	ROLAND GARROS
G_tourney_name	1	tourney_name	ROME MASTERS
G_tourney_name	4	tourney_name	ROTTERDAM
G_tourney_name	4	tourney_name	SANTIAGO
G_tourney_name	0	tourney_name	SAO PAULO
G_tourney_name	5	tourney_name	SHANGHAI MASTERS
G_tourney_name	0	tourney_name	STUTTGART
G_tourney_name	1	tourney_name	TOKYO
G_tourney_name	6	tourney_name	TOUR FINALS
G_tourney_name	1	tourney_name	US OPEN
G_tourney_name	2	tourney_name	WIMBLEDON
G_surface	0	surface	CLAY
G_surface	1	surface	GRASS
G_surface	1	surface	HARD
G_tourney_level	1	tourney_level	A
G_tourney_level	0	tourney_level	D
G_tourney_level	2	tourney_level	F

G_tourney_level	0	tourney_level	G
G_tourney_level	1	tourney_level	M

C. Acceso a repositorio online con el código utilizado

A través del siguiente link, se puede acceder al repositorio online que contiene el código (SAS y R) utilizado en este trabajo, tanto para el Objetivo I (predicción de la duración de un partido de tenis), como para el Objetivo II (elaboración de un modelo explicativo para los partidos de Rafael Nadal):

o bien:

https://github.com/alexrh1/TFM_AlejandroRuiz-